LINUX VOICE

The magazine that gives back to the Free Software community

April 2014

114 PAGES OF NEURAL ENHANCEMENT!

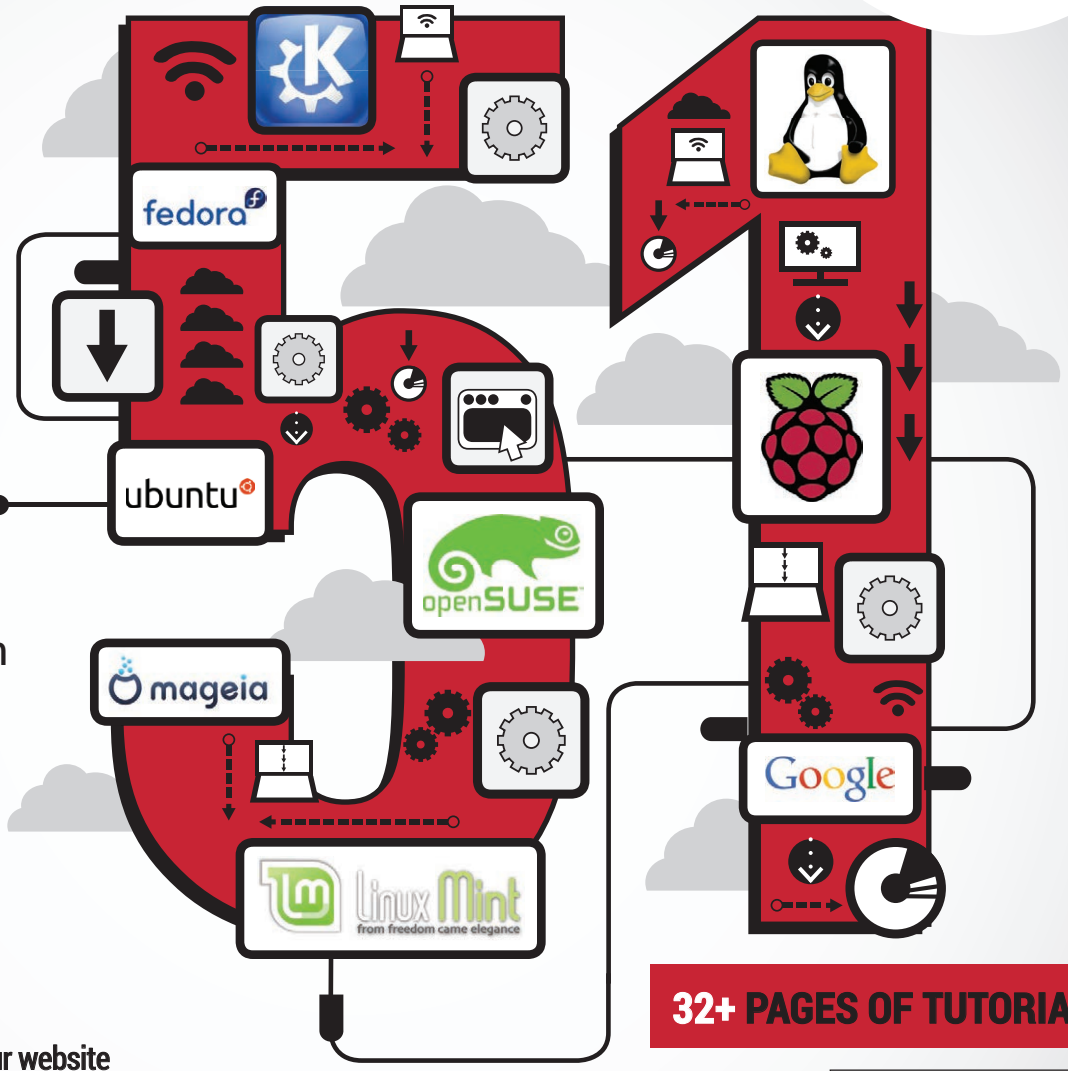
PRIVACY
PGP
Stop President Obama from reading your emails

RASPBERRY PI
BREWPI
Brew fine ales with free software. Mmm, beer...

YE OLDE CODE
LOVELACE
Tonight we're going to program like it's 1843

THE BEST FREE SOFTWARE 2014

Discover the 51 best things about free software right now with our ultimate roundup



FREE SOFTWARE | FREE SPEECH

32+ PAGES OF TUTORIALS

PYTHON Pipe live data into your website
SYSADMIN Secrets from the server room revealed
BITCOIN Get your head around the gold bullion of the internet

REVIEW
MAGEIA 4
The KDE 4 desktop is now a beautiful swan

DON'T BE EVIL
OWNCLOUD 6
Free yourself from Google's tentacles

ISSN 2054-3778
9 772054 377001 04
April 2014 £5.99 Printed in the UK

As the stewards of the Open Source Definition (OSD) and the community-recognized body for reviewing and approving licenses as OSD-conformant, the OSI faciliates Open Source community-building, education, and public advocacy to promote awareness, adoption and the importance of non-proprietary software.

The OSI, with global reach, champions Open Source software and projects, meeting with developers, users and communities as well as with executives from the public and private sectors to explain how Open Source technologies, licensing & models can provide economic, strategic and societal advantages.

Open Source Initiative

welcomes

Linux Voice

to the global Open Source community.

Join the Open Source Initiative now
and be a part of the future of Open Source.
<opensource.org/members>



ELECTRONIC FRONTIER FOUNDATION

Help EFF Defend Your Rights in the Digital World **eff.org/join**

Friends, geeks, hackers

The April issue

LINUX VOICE

Linux Voice is different. Linux Voice is special. Here's why...

1 At the end of each financial year we'll give 50% of our profits to a selection of organisations that support free software, decided by a vote among our readers (that's you).

2 No later than nine months after first publicaton, we will relicense all of our content under the Creative Commons CC-BY-SA licence, so that old content can still be useful, and can live on even after the magazine has come off the shelves.

3 We're a small company, so we don't have a board of directors or a bunch of shareholders in the City of London to keep happy. The only people that matter to us are the readers (you again).



GRAHAM MORRISON

A free software advocate and writer since the late 1990s, Graham is a lapsed KDE contributor and author of the Meeq MIDI step sequencer.

As I write this, it's a sunny Saturday morning and I'm sitting in the kitchen. The issue is 90% finished and we've got eight hours until the PDFs need to be at the printers. We're used to crunch times like this. Stacey is adding the barcode to the cover. Andrew is working his magic. Mike is organising our online store while Ben polishes the new website. It has been sometimes difficult, sometimes challenging but always absolutely wonderful. We've created the magazine we always wanted to create, which we hope best represents the Linux and Free Software communities.

Which is why there's no better place to start than with a celebration of the very best that Free Software has to offer. For our cover feature, (**p38**), we chose 51 of the most awesome projects, but we could have chosen hundreds. The extent of open source penetration is staggering, and it's only going to become increasingly influential. The next few years will be the most exciting and revolutionary yet – all of us at Linux Voice can't wait to start making our own modest contribution.

Graham Morrison
Editor, Linux Voice

SUBSCRIBE ON PAGE 36

THE LINUX VOICE TEAM

Editor Graham Morrison
graham@linuxvoice.com

Deputy editor Andrew Gregory
andrew@linuxvoice.com

Technical editor Ben Everard
ben@linuxvoice.com

Editor at large Mike Saunders
mike@linuxvoice.com

Malign puppetmaster Nick Veitch
nick@linuxvoice.com

Creative director Stacey Black
stacey@linuxvoice.com

Editorial contributors:
Mark Crutch, Liam Dawe, Juliet Kemp, John Lane, Vincent Mealing, Simon Phipps, Jonathan Roberts, Mayank Sharma

What's hot in LV#001



ANDREW GREGORY

She doesn't get the credit she deserves, so it's an honour to have a tutorial on Ada Lovelace's work with the Analytical Engine. **p88**



BEN EVERARD

For me, it's the delicious fusion of hardware hacking, the Raspbery Pi and quiet inebriation promised by Graham's BrewPi guide. **p76**



MIKE SAUNDERS

FOSDEM: so many passionate, inventive and inspiring geeks under one roof recharged my passion for Free Software. **p30**



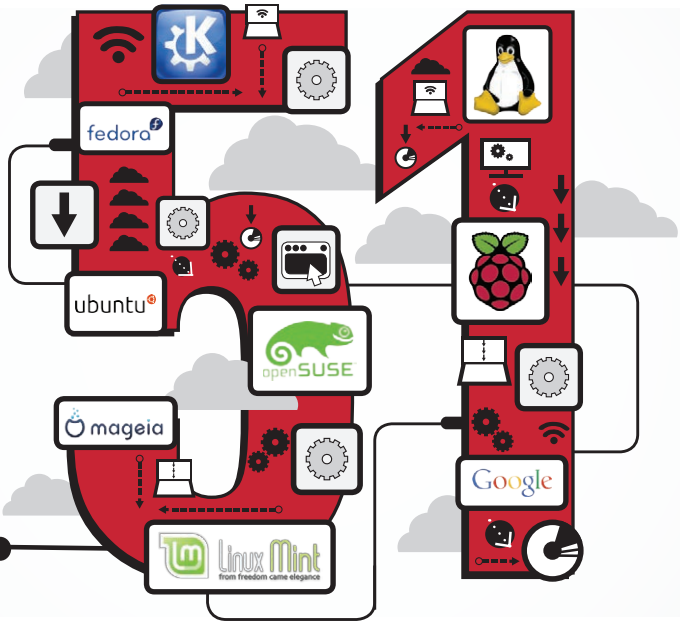
CONTENTS

April LV001

We are all in the gutter, but some of us are looking at the stars

SUBSCRIBE ON PAGE 36

38 THE BEST FREE SOFTWARE 2014

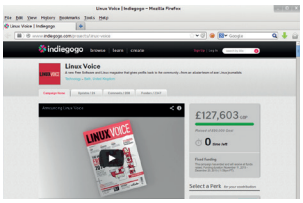


Discover the 51 best things about Free Software right now.

50 BITCOIN

What in the name of Zeus is going on with this unregulated digital currency?

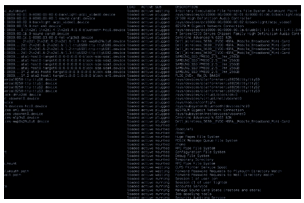




56 LINUX VOICE
The mad journey we've been on so far.



60 FAQ Wayland: what is it, and why should we even care?



62 SYSADMIN
Jon Roberts on the brave new init world.

REGULARS

06 News
To paraphrase Marvin Gaye: what's going on?

08 Distrohopper
The latest comings and going on planet distro.

10 Gaming
Yes! Linux is a serious gaming platform, at last!

12 Speak your brains
Because 'letters' is an archaic term for this bit.

16 LV on tour
We go where others fear to tread. Its name? Blackpool.

30 Group test
Find the best encryption tool for you.

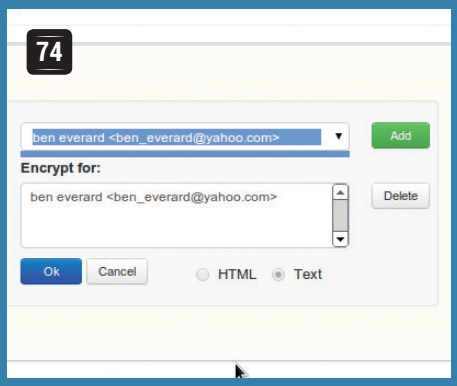
46 Interview
Florian Effenberger, on forking OpenOffice.

106 Masterclass
This issue: say hello to Vim and Firefox.

112 On your DVD
What's inside your free monster 8GB disc.

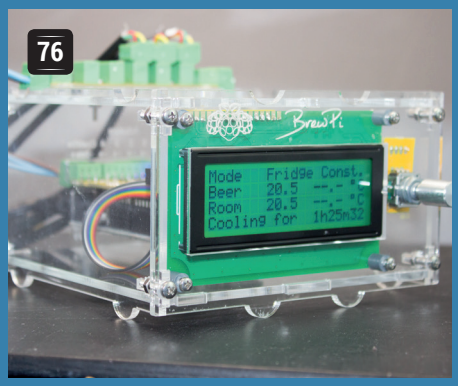
114 My Linux desktop
The bodger of Crunchbang show us his man cave.

TUTORIALS



PGP: Keep your messages secure

Encrypt your emails and reclaim a little bit of privacy from Big Brother/the NSA/GCHQ.



BrewPi: Brew beer at home with a Raspberry Pi

Control and monitor the brewing process with the help of a handy Linux-powered kit.



OwnCloud: Say goodbye to Google Docs & Gmail

Set up your own cloud services and get the convenience of the cloud without the intrusive ads.



Old code: Ada Lovelace & the Analytical Engine

Travel back to the dawn of time to see how programming began – then try it for yourself.



Arch Linux: Installation and setup made easy

Stay effortlessly* up to date. *Some effort required



Python: Build dynamic web pages

Pipe live data into your website.

100 Code 101: Euclid's algorithm

Learn Python from the Ancient Greeks.

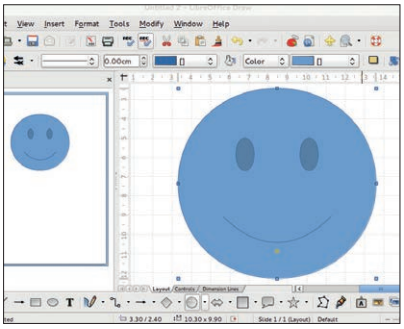
102 Bash: Fun with regular expressions

Trust us: you can enjoy using Grep.

REVIEWS



22 **Mageia 4** If you're a desktop magpie, constantly distracted by new shiny things, this user-friendly Linux flavour is for you.



24 **LibreOffice 4.2** Better than Microsoft Office? You bet your life it is – the #1 office suite for Linux is more than just a clone.

25 **Lightworks Pro 11.5** Video editing on Linux just got a hell of a lot better. Now prepare for your close-up, Clooney.

26 **FreeBSD 10** We've always had a soft spot for this stable Unix derivative. You see, Linux is not the only fruit…



27 **Google Nexus 5** We've lived with the Nexus 5 for three months – is it worth paying 300 Imperial Credits for?

28 **Books** From the Arduino to OCaml to feudalism, there's something for everyone in the Linux Voice library.

NEWSANALYSIS

The Linux Voice view on what's going on in the world of Free Software.

Opinion

"...and the filters don't work/ they just make it worse..."* *sincere apologies to The Verve

Web filters to protect children from dodgy online content don't work. Here's why.



Simon Phipps is president of the Open Source Initiative and a board member of the Open Rights Group and of Open Source for America.

The UK government has pressured ISPs into applying content filters to their customers' connections, in the name of protecting children from unsuitable content. During 2014, ISPs will be approaching their customers and trying to persuade them to turn on filtering. But this is a mistaken approach arising from magical thinking – "this thing should exist so it must be possible". Content filters can't work, for several reasons:

For the most part they can be avoided. Techniques such as using a freely-available VPN tool such as TunnelBear, or switching to non-ISP DNS enable users to effortlessly route round filters. As a consequence, relying on filters to do your parenting for you is foolish. Not only are they no substitute for parental oversight and care, they inculcate a careless reliance.

They attempt to make objective a task which is subjective. For example, some people will regard websites promoting gay rights or giving information about abortion as unsuitable, while others will treat both topics as essential resources. Who gets to decide for us all? The answer in most cases is "nobody knows", since the ISPs are largely buying the blocking facility from third party suppliers rather than building it themselves. Statements by filtering advocates take it as read that there's a consensus on what's bad are deceptive.

I'm sorry Dave, I can't do that

The government proposes a whitelist of sites that should never be blocked, but this approach is flawed too; partly because their vision of which sites should be whitelisted only includes obvious, politically appealing cases like child welfare charities, omitting to mention harder cases such as mutual-support groups, political comment and satire and completely ignoring the sort of free speech cases that are politically unappealing to the government. Indeed, a comprehensive whitelist is probably impossible, because the internet, like space, is vastly, hugely, mindbogglingly big.

Even if a whitelist could work, most content providers won't know they should be on it as blocking is invisible to them. Since the filter service is applied by ISPs to their customers' connections as part of the service, they rely on customers raising the alert on overblocking. As a web page provider, I have no way to know whether a given ISP is blocking my site, and when I eventually find out there's no deterministic way to get it fixed since neither the ISP nor their third-party provider have any duty to help me; in fact, the ISP's contract with their supplier may actually prohibit them from helping me.

Most content providers might not even think to check anyway, even if there were a way to do so. Recently, the jQuery website was added to the block list for UK ISP Sky after the domain was mistakenly listed in the "malware and phishing" category. This unexpectedly broke many websites, since over three-quarters of the top 10,000 websites use **jquery.com**-hosted components. A church website in Sheffield was blocked; my own company website was blocked. The robots that do the ranking can take potshots at pretty much anything, and only the customers of the ISP involved could ever know.

Think of the children!

Meanwhile, parents are lulled into a false sense of security. The web is something of a mystery to many, and the assurance that "parental control filters" are keeping their children safe may well reduce the urgency of understanding how to supervise children on the web. The correct path is sitting with children, assisting their use of technology, explaining how to decide who to trust, explaining when to ask for help or permission, applying discipline wisely.

If it were possible to magically determine the suitability of any random website for any random web user, and if filtering could be made uncircumventable without destroying the utility of the internet, maybe it would be OK to have a censorship switch that parents could flip. But none of that is possible, and the facilities we're being sold will do more harm than good. You can help; check out **blocked.org.uk**. That will help practically, and also fuel the political battle.

"The government proposes a whitelist of sites that should never be blocked, but this approach is flawed."

SUSE • Firefox 27 • Valve • Tizen • Developers • OpenSSH • Chakra Linux • Kernel

CATCHUP Summarised: the nine biggest news stories from the last month

1

SUSE announces live kernel-patching system

Imagine being able to patch your kernel without having to reboot. And imagine the staggering uptimes you'd be able to achieve... A few methods have been developed to do this, but they have never made it into the mainstream kernel. Now the SUSE team has announced kGraft, with a first release due in March. **http://tinyurl.com/susekgraft**.

2

Firefox 27 released

Given the break-neck pace of development in Firefox, it's hard to keep track of all the new features. Firefox 27 enables you to run more than one service at a time with its Social API (so you can get notifications from multiple sites), and also enables support for TLS 1.1 and 1.2 by default. SPDY 3.1 is included too.

3

Valve offers free games to Debian developers

SteamOS, Valve's gaming platform based on Debian, could massively shake up the computer games market. Users will be able to play Valve's ever-growing range of titles without having to even boot Windows. A healthy relationship between Debian and SteamOS is important, so Valve is now offering free subscriptions to Debian developers.

4

First Tizen smartphone leaked

We've seen a bunch of Linux-based mobile platforms come and go – Maemo, Moblin and Meego – and now we have Tizen. The Samsung ZEQ 9000 will be the first phone to run this operating system, sporting a 4.8-inch display and a 2.3GHz processor. Physically, the phone looks a lot like the current Galaxy range; it's due for launch later in the year.

5

80% of kernel developers are paid

Every year the Linux Foundation tracks changes to the Linux kernel source tree, and determines how much code is being contributed by commercial companies. Newly released stats show that over 80% of developers are being paid to work on the kernel, with Red Hat, Intel and Texas Instruments the three biggest contributors.

6

OpenSSH 6.5 released

You'd think OpenSSH is secure enough already, what with it being a product from the OpenBSD camp, but given the NSA spying revelations you can't be paranoid enough. Version 6.5 includes new cyphers and key types, to reduce even further the chance of someone peeking at your remote login sessions.



7

Chakra 2014.02 available

With most distros you upgrade once or twice a year and get a bundle of new stuff. There are also rolling-release distros such as Arch, which give you a constant stream of new software. Chakra uses a "half-rolling release" model: the base system is updated only after extensive testing, whereas new desktop apps flow in all the time. **www.chakra-project.org**

8

Kernel 3.13 brings new packet filter

One of the biggest new features in kernel 3.13 is nftables, the successor to iptables. This is a "packet classification framework" – ie, a system for choosing where network packets go, as used in routers and firewalls. Nftables compiles rules down to pseudo-bytecode, and promises to make life easier for administrators thanks to a simpler syntax.

9

2014: the year of code

After more or less being shamed into action by the efforts of Code Club, Young Rewired State and the Raspberry Pi, the UK government has launched something called the Year of Code, which is supposed to encourage more people to learn to program. According to the website, 1 in 6 adults lack digital skills, though what this actually means is anyone's guess.

DISTROHOPPER

We've tapped GCHQ's communications to find out what's going on in distro land.

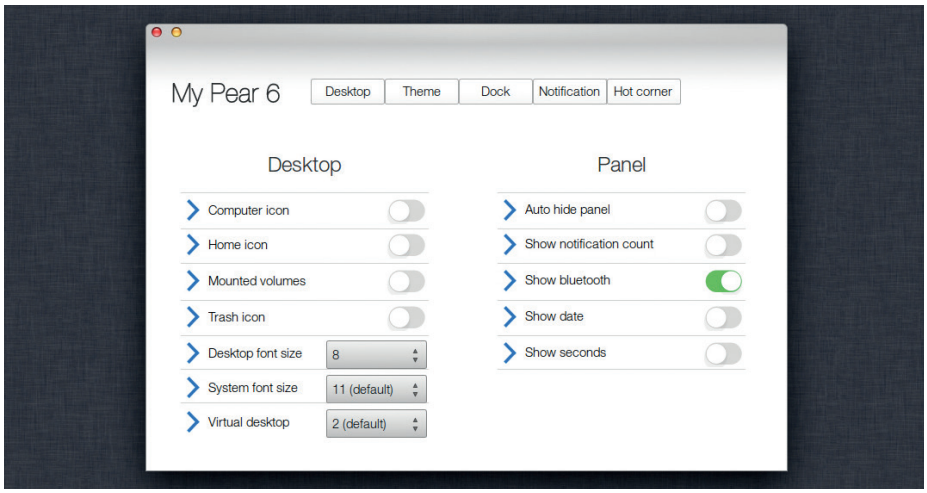
The end of the road for Pear OS

This slick desktop OS takes its final bow. Or does it?

Pear OS never really hit the big time, but it still managed to court more than it's fair share of controversy. Fans adored its slick good looks and ease-of-use (both heavily influenced by Apple's OS X), while critics pointed out that it was little more than a tweaked version of ElementaryOS with an icon set that looked suspiciously similar to Apple's.

Late January, Pear OS disappeared. The website switched to a holding page saying that a "very large company" had bought Pear OS, and that its future was in the hands of its new owner. The downloads were taken offline, and anyone with data stored on the Pear OS cloud servers had 10 days to recover it before it was deleted. By the time you read this, more may have been revealed at **http://pearlinux.fr**.

About a week after Pear's mysterious disappearance, Clementine OS appeared at **www.clementineos.hj.cx** promising to continue PearOS. It's not clear exactly how they'll do this, or if they have access to Pear's source code that others haven't been able to get. It'll be a while before we find out though, since the first version isn't scheduled to be released until after Ubuntu 14.04 comes out in April. In the mean time, the Clementine project is hosting the Pear OS ISO images, though we wouldn't recommend people install a distro that won't be supported.



Pear Linux faced accusations that it wasn't sharing its source code, as required under the GPL.

For people now at a loss with the passing of Pear OS, or who like the screenshot, we humbly suggest you take a look at Elementary OS. It's less Apple-like than Pear, but still has a similar look and feel. Not only that, but it's a much better open-source citizen. Not only does it they produce its own desktop environment, but it's also helping in the development of the Shotwell photo manager. Elementary OS also has a substantial development team, and doesn't just rely on a single person, so its unlikely to disappear overnight.

STEAMOS AND THE COMMUNITY

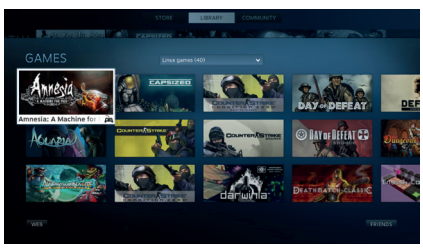
Valve's gaming distribution has dominated the news recently, and we won't rehash everything here. Instead, we're going to talk about some smaller aspect of development. The original installer was a bit of a pain to use and required UEFI. This wasn't a huge problem, because it was only a beta release aimed at experts.

However, as it's built on Linux, the whole platform is open for tweaking. A group of users including directhex and ecliptik took to github to create a new version of the OS with support for non-UEFI devices, DVD installing, and other basic features. They called their project 'Ye Olde SteamOSe'. Valve has now pulled these improvements into a new version of Steam OS, making it easier to get up and running.

In further Steam news, Valve has provided all Debian developers with free access to all Steam games. Debian is known for its strong adherence to the principles of free software, so it's not clear how many of the developers will take up this offer of commercial software, but we think it's a nice gesture either way.

How free is Valve?

Valve is a relative newcomer to the Linux world, and its background in proprietary software has led some people to question how well they'll fit into the open source ecosystem. It's too early to tell exactly, but the early signs are about as favourable as they can be for a proprietary software company. It's worth remembering that Valve's history lies firmly in the PC game modder community, which shares a tinkerer philosophy with the open source and free software community.



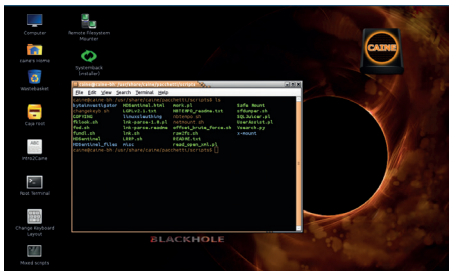
There are 302 games available for SteamOS, and it hasn't even made it out of Beta yet.

System Rescue CD & Caine

Not a lot of people know about these super rescue distros.

Hardware failures are always difficult, but they're a lot less difficult now than they were a few years ago, thanks to the advent of live distros. Almost every distro now comes with a live version that can be put into use should your hard disk fail, or some other calamity befall your system. However, there are a few Linuxes that are specially designed to help in this case, and two of the more popular got an update earlier this year: System Rescue CD and Caine.

The first of these is pretty much the standard in rescue discs, but the latter has some interesting uses. Caine is designed for computer forensics -- think CSI but with computers. There's a large cross over between this and extracting information from a breaking system. While System Rescue CD remains our favourite for most repair jobs, Caine comes into its own when you're more interested in pulling data off a system than repairing it. It's also got some useful tools for investigating phones.



It's worth getting to know your way around a rescue distro before you have a system failure, so you'll know what to do when things go awry.

Gobo: rethinking the file structure

Everything you know about the Linux filesystem is wrong.

The Linux filesystem is more or less the same in just about every distro. This means you can switch between them and still know where to find things. You'll even find your way around BSDs and Unixes without any major problems. Is this standardisation holding back the evolution of the system though? Does it still make sense to split bits of programs up into **/bin**, **/etc** and **/usr/share**? Is **/opt** really useful? Is **/var** still relevant with modern hardware? The answer to all questions may well be yes, but can you really be sure without trying out some alternatives? The answer to that is definitely no.

Gobo Linux is a distro that's thrown out the old filesystem paradigm and invented its own. The root directory has six folders: **/Depot**, **/Files**, **/Mount**, **/Programs**, **/System** and **/Users**. Perhaps **/Programs** is the best example of how it differs. It doesn't use a package database for package management because it doesn't need one. The file hierarchy holds all the information. For example, the program bash is held in **/Programs/Bash/<version-number>/** where there's a separate folder for each installed version. **/Programs/Bash/Current** is a link to the folder containing the latest version. Packages, then, are just archives that can be unpacked into this structure. Alternatively, any software you compile yourself can simply be copied into the correct place.

The project has just sprung back to life, and is working on its first new version since 2008. We applaud this bold experiment!



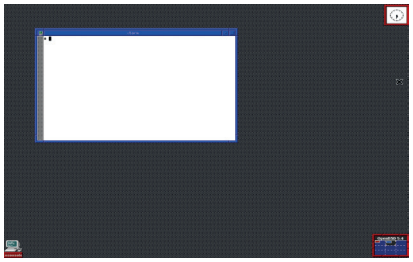
GoboLinux wouldn't start on our hardware, but ran fine in Qemu.

OPENBSD AND CASH

Asking 'How much money does it take to create a distro?' is a bit like asking 'how long is a piece of string?' Both can be as little or as long as you like. If you really wanted, you could create a new distro based on something like Ubuntu or Fedora in a weekend with no budget, but it probably wouldn't add much to the already existing sea of distros.

On the other hand, the OpenBSD foundation is hoping to raise US$150,000 this year (that's about £91,000 in old Imperial money). $20,000 is needed just for electricity in the build farm. This all has to come from donations since the foundation doesn't have any commercial interests. It looked like it may be the end of the road for this security-centered Unix when, late last year, a campaign to raise funds for the power bill failed. Early this year, though, a final donation drive caught the attention of the internet and $100,000 flowed into the foundations coffers in just the first three weeks of the 2014 with donations from Google, Facebook and the MPEx Bitcoin Securities Exchange amongst others.

This is good news not just for BSDers, as OpenBSD projects – most notably OpenSSH – are included in most Linuxes. The work by Theo de Raadt and the rest of the team help keep us all safe regardless of the kernel we use.



OpenBSD's may not be pretty, but it is secure (and other desktops are available).

GAMING ON LINUX

The tastiest brain candy to relax those tired neurons

LINUX FOR THE WIN!



Liam Dawes is the brains behind www.gamingonlinux.com, the home of Linux gaming on the interweb.

Linux has long been a viable platform for gaming, but it is only now that mainstream gamers are starting to sit up and take notice. Despite what many people may think, the thanks for this shouldn't solely go to Valve for using Linux as the basis of its Steam OS: there's a long history of gaming on the Linux platform.

Id Software was one of the first major names to bring its games to our little platform, starting with Doom and ending with its latest Linux port of Enemy Territory: Quake Wars. Id's involvement doesn't end there though – it also open sourced many of its game engines, which have given rise to some of Linux's best-known first person shooter, such as Xonotic.

Not just Valve

The flow of indie games coming to Linux prompted a leading game store to open up to Linux and push out the Desura Linux client, which was itself open sourced enable others from the community to help out; even though licensing issues meant that the number of contributors never really took off, this was progress.

And we now have Steam, the behemoth of PC gaming. Who would have thought that Valve would bring its AAA games to Linux?

My question to you, readers, is: will Linux ever overtake Apple to become the second biggest desktop operating system? With Steam Machines using the Debian Linux based SteamOS it could finally be a possibility.

Liam Dawes, gamingonlinux.com

0 A.D.

Battle in real time strategy like an Egyptian.

Like an overdue Christmas present the team at Wildfire Games has released the fifteenth alpha version of the epic open-source real time strategy game 0 A.D.

The latest alpha version added some really essential features that were previously missing, including a multi-player lobby, so it should now be a total cinch to find people to play online.

In addition to being easy to play online there is yet another playable civilisation – the Ptolemaic Egyptians – with unique units and gameplay features included for them.



A new army is readying for battle!

0 A.D. aims to be of AAA production quality, which shows in abundance. It was originally closed source, but the team opened it up with the aim of gaining new members, and this strategy has paid off. They have run several successful crowdfunding, and are now using it to fund programmers to make 0 A.D. even better!

http://play0ad.com

THE BATTLE FOR WESNOTH

A seriously slick open-source turn-based strategy.

Cheering the cold, dark winter months considerably, the team behind The Battle for Wesnoth have released a new version of this well known turn-based strategy adventure.

They have further tweaked the UI of the game to make it more playable, including an overhaul of the mouse interaction in the game, and the Khalifate faction, which doesn't use magic, is now playable in more multi-player areas too.

Battle for Wesnoth has some great single-player gameplay alongside online multiplayer too, so you can play your friends across the world. It has been around a few years now, but that doesn't mean it hasn't aged well! It is updated all the time with new features, tweaks and bugfixes to keep it fresh.



It's like *Game of Thrones* without Sean Bean.

It's currently Stable, meaning game-breaking bugs should be very hard to find, so feel free to really enjoy this one.

www.wesnoth.org

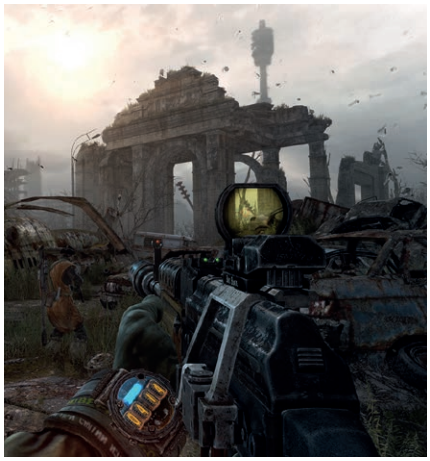
METRO: LAST LIGHT

The most graphically intensive first-person shooter on Linux!

Graphics cards of a nervous disposition, look away now: what we have here is probably the most demanding, hardware-punishing game on Linux. You won't be surprised to learn that it's a first-person shooter, from 4A Games called Metro: Last Light, which promises to simultaneously cooing with delight at its prettiness while screaming with terror when an enemy jumps across your screen.

4A Games is a Ukrainian studio, and Metro: Last Light draws on the the work of Russian novelist Dmitry Glukhovsky, particularly his novel *Metro 2033*. Set in a bleak world that has been ruined by nuclear weapons, the game forces you to rely on your wits as much as brute force, with limited ammunition forcing players to think rather than just run around shooting at things.

While we Linux gamers don't have the prequel game (also called Metro 2033) just yet, it will hopefully only be a matter of time now that we have 4A Games bringing its catalogue to Linux.



We don't care if the future's an apocalyptic wasteland, as long as it looks as nice as this.

http://store.steampowered.com/app/43160

CORSIX-TH

Bullfrog's classic hospital sim brought back to life. Ooh, matron!



Remember Theme Hospital from Bullfrog? Well Corsix-TH is an open source engine for playing the comedy hospital simulation game natively on Linux, and it recently released version 0.3 which further enhances the engine.

While the engine is free, you have to purchase the game assets from a site such as **GOG.com**. This is one of the awesome things about the Linux and open source community – rather than being forgotten, games that we loved when we were younger get revived with brand-new features.

It may be in beta, but Corsix-TH is fully playable and extremely fun.

https://github.com/CorsixTH

OPENMW

Massive open-world RPG engine brings Skyrim predecessor.



The post-Christmas presents haven't stopped yet folks! There's another open source game engine with a brand-new release: this time it's OpenMW, which brings us The Elder Scrolls: Morrowind RPG. This new version includes probably the most important features, including combat AI and spell casting, including visual effects for magic.

Much like Corsix-TH, OpenMW is an open-source engine designed to let you run Morrowind on your favourite open source operating system. It's undergoing heavy development and missing some essential features from the original, but it's well worth keeping an eye on.

https://openmw.org/en

ALSO RELEASED...



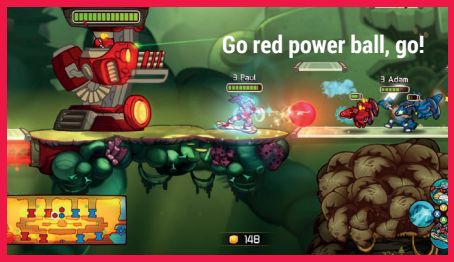
Faster Than Light

Faster Than Light has announced a free DLC expansion pack to this infuriating game. Included in this free update are new ship layouts, and a brand-new alien race called The Lanius, whose members don't need oxygen to live. Not only that, but they will suck oxygen out from any room that they are in. Scary! www.ftlgame.com



Dota 2

Valve has graced us with another great free-to-play title: Dota 2. A multiplayer online battle arena (MOBA), Dota 2's Linux version includes some pretty nifty features not available on other platforms, as well as a pop-up notification when a game is ready to play – very handy for serial Alt+Tabbers. http://store.steampowered.com/app/570



Awesomenauts

There's an expansion pack to Awesomenauts, titled Starstorm, which includes new playable characters, game replays and custom game-modes. This is another MOBA although it's far simpler than Dota 2 and plays like a platformer. The community is a little friendlier, due to it being a simpler game to get into. http://store.steampowered.com/app/204300

LINUX VOICE YOUR LETTERS

Got something to say? An idea for a new magazine feature?
Or a great discovery? Email us: letters@linuxvoice.com

LINUX VOICE STAR LETTER

THE FIRST TASTE OF FREEDOM

I remember when I finally ditched Windows. At last I'd broken free from the XP safety net, but it felt a little bit like OS freefall. Could I really manage without a corporate-backed product? Could I genuinely rely on a community of enthusiasts when something went wrong? Would I cope without the comfort and security of a familiar brand in a new world where things weren't the same and sometimes broke? It didn't take long before my fears were dispelled and I found that the gamble had paid off.

Then I remembered all the headaches and issues I used to have with Windows which lead me to this decision in the first place. If any of the above sounds strangely familiar territory with your latest publishing venture, then fear not. You are still among friends and we have made this journey together, for the better.

Chris Sandles



Who needs Microsoft for technical support? Linux fans at user groups help for free (photo credit: http://tinyurl.com/plz7tfp).

Graham says: Well said Chris. The Linux and Free Software community may be incredibly passionate and argue a lot, but it's also the most helpful community I've ever known.

ON A ROLL

I have two computers here, one of which is running Mint. I have installed every version since Maya, the LTS release, and having a separate home partition it has been no problem. But, with each install, I have had to re-download all the added programs from the repositories. Not an intuitive task for at least two people I have persuaded to try Linux.

Last year I came across SolydXK, ran the KDE version in VirtualBox, and was impressed by the way everything was updated every month. On behalf of novice users, would it be better to have a rolling release, or stick with an eighteen month-ish distribution?

Richard

Mike says: Novices are better off with the clearly defined releases, I think. Rolling distros are good for living on the bleeding edge, but major kernel/X/library updates can cause unexpected, hard-to-fix breakages.

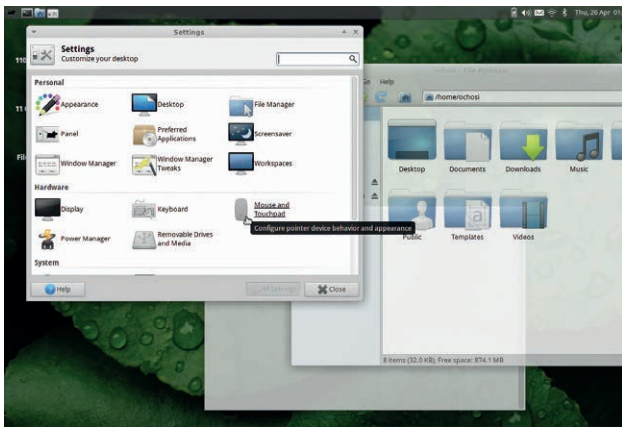
XFCELLENT

It would be great if your magazine could review a few more Xfce distributions. I noticed Xfce often gets overlooked in comparison to bigger desktop environments such as Gnome and KDE.

I was hoping you guys could do an a round up of several Xfce distributions and write a comparison for your readers. Looking forward to your magazine. I'll be getting my digital subscription soon.

Alvin Wilson

Ben says: Thanks for the feedback Alvin. You're right that Xfce doesn't get as much attention as the "big" desktops, but the times are a-changin'. We've seen a large bunch of former Gnome fans move to Xfce after the Gnome 3 redesign, as it provides a lot of what Gnome 2 used to: a familiar-looking, reliable GTK-based desktop environment. We'll keep an eye on Xfce-based distros; Xubuntu is the most notable, but we always like to hear of others. If you discover one, let us know!



Xfce has come a very long way since its days as a clunky-looking clone of CDE (the Common Desktop Environment).

BANGIN' CHOONS

I wonder if readers would like to share their favourite songs? I love anything by GNU Kids on the Block, and we should all be grateful for the contribution that Kernel Parker made to modern music, but I think my all time bests are Cliff Richard's "She's a /dev/null Woman", Buddy Holly's "Wake Up Little SUSE", Paul Young's "Wherever I Lay my Red Hat" of course, and Phil Collins' breathtakingly un-annoying "Su Sudo". In fact I like any cd.
Owain Clarke



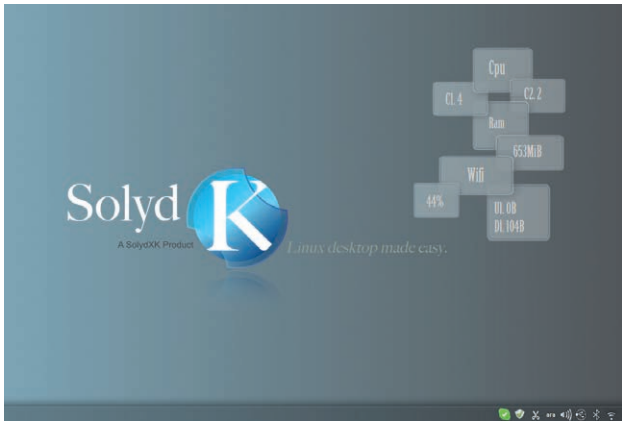
Andrew says: All groovy beats, but let's not forget the classic "Stand by your man(page)". If anyone can somehow fit Slackware into a song title, let us know...

Stallman can sing too: www.youtube.com/watch?v=9sJUDx7iEJw.

DISTRO HUNT

I've been using Ubuntu for quite some time, installed 13.10, and got quite disappointed. So I had to look for a new distro. After a little searching the net, I found a young KDE distro, based on Debian, inspired by Mint Debian edition, does monthly update-packs (semi-rolling releases), looks great, has the latest and greatest, comes with Firefox, Thunderbird, LibreOffice... I think I found a new home: SolydXK!
Sofie Struyve

Mike says: Wow, that's two mentions of this distro in the same letters page! SolydXK's relatively conservative approach to the rolling release model (bundled monthly updates) may help to avoid the occasional breakage that can afflict other rolling distros. If anyone else wants to check it out, the website is **www.solydxk.com**.



SolydXK: not well known now, but could be a big distro soon.

THE CODE AHEAD

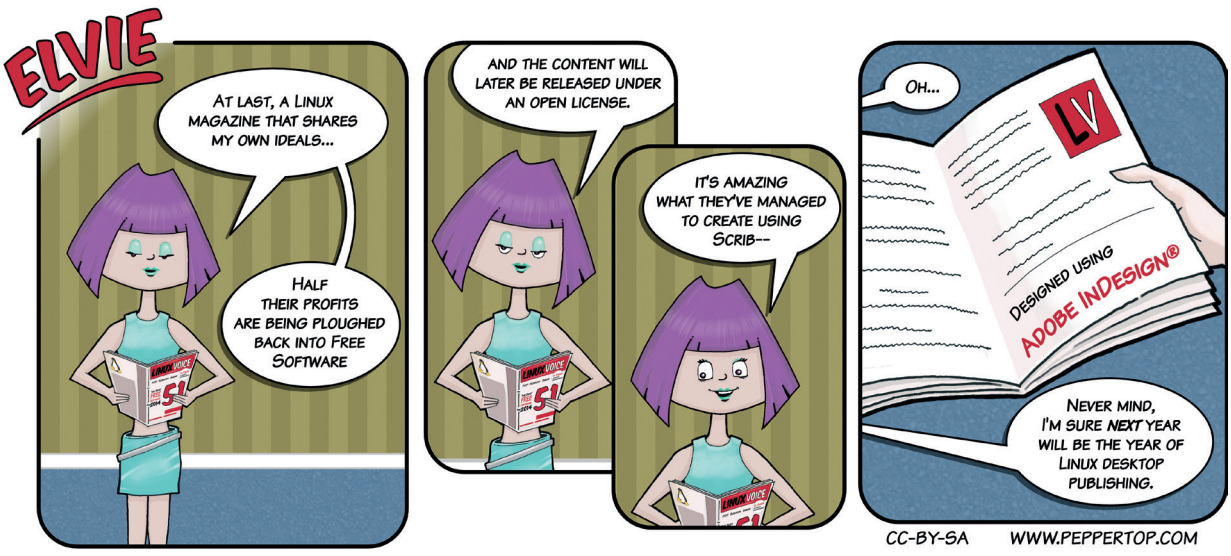
Just a note on Linux and FOSS in general. I made the big move about five or six years ago. Since that time I started coding again and have really started to enjoy computing again. In short, computing is fun again.

I hope that you continue with coding tutorials in Linux Voice. They are great for those who have an itch, but don't where to start, and for those of us who have a favourite language, but don't like change. We get to see how the other language is structured, which means we can then better read source from other languages.

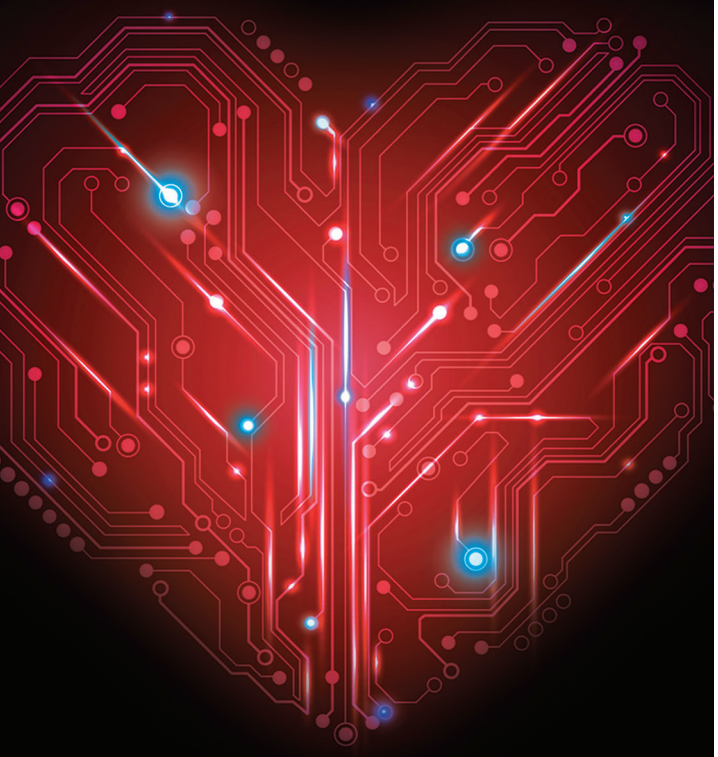
Do keep up the good work we have all seen from you in the past.
Bill Palone

Mike says: Thanks for the feedback Bill. Hopefully you'll be glad to see a regular coding section in Linux Voice, and in next month's cover feature we'll have programming projects for newbies as well as experienced hackers.

I totally agree that coding makes computing fun, and it's great that the Raspberry Pi is encouraging kids in schools to learn the art – apart from helping kids out in the job market, programming skills help language, maths and logical thought. One of my next projects is to port MikeOS to the Pi, but given that it's all written in x86 assembly language and the Pi works on an ARM chip, that'll be a truly epic job...



WHEN IT'S CRITICAL...



WE WILL KEEP YOUR DATABASE ALIVE

2ndQuadrant IS PROUD TO SUPPORT LINUX VOICE'S LAUNCH EDITION

2ndQuadrant's Platinum Production Support for open source PostgreSQL provides a guaranteed 15 minute (human) response, 4 hour workaround and guaranteed bug fix within 24 hours. 24 hours a day, 365 days a year.

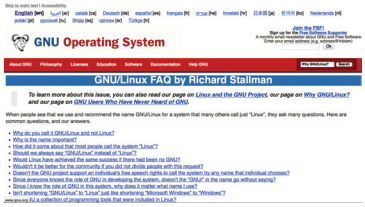
We'll keep your business alive.

2ndQuadrant
Professional PostgreSQL

+44 (0)870 766 7756
2ndquadrant.com/support

SATNAV SIGHS

Linux Voice! I'd like to express my absolute disgust that your magazine isn't called GNU/Linux Voice. I will be writing to my MP about this very important... Only joking. Good luck with the magazine, and I hope you do give credit to the GNU project when appropriate, even though I think the naming debate is rather overblown.
Dan Konovalov



The GNU/Linux naming FAQ weighs in at over 9,000 words. Nomenclature is a serious business indeed...

Andrew says: Rest assured, the GNU/Linux FAQ on **www.gnu.org** is my bedtime reading material. One day I'll get to the end... In all seriousness, GNU, Richard Stallman and the Free Software Foundation have contributed hugely to the operating system we use and love today. Big thanks to them. But there must be a prettier name (and "Lignux" isn't it).

GNU'S TALKING?

TomToms are very annoying; they may use a Linux kernel to work but to update them you require a Windows or Apple computer. Their update program just will not work with a Linux Box, even when running with Wine.

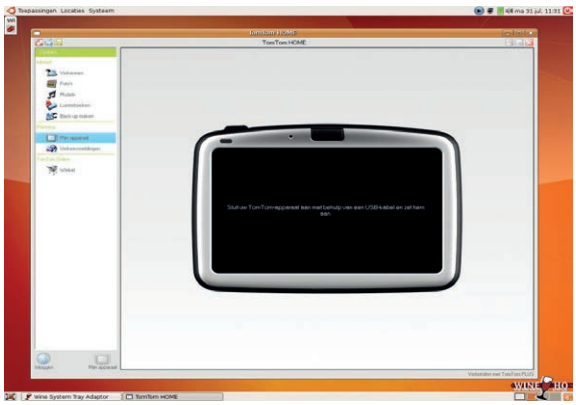
Plugging the satnav into the USB socket on my Linux Mint computer allows me to see inside the TomTom just as if it were an attached disk. This is all fine and dandy but, unless you can download the appropriate files from TomTom and then know what to do with them, it all becomes a bit pointless.

Do you have any ideas? Is there such a thing as a Linux-friendly satnav on the market in the UK? And good luck with the new magazine – I'm really looking forward to the first (and subsequent) editions falling through my letter box.
Ray Woods

Mike says: I'm not au-fait with sat-nav systems, as I navigate by the moon, the stars, and ancestral memory, but maybe one of our readers can write in with a recommendation. Or you could perhaps add your name to the "Tom Tom Linux Support" petition online: **www.petitiononline.com/tomlinux**.

And yes, there is something very silly about devices that run Linux, but don't actually integrate well with desktop Linux installations. It seems like manufacturers are missing an opportunity for a big, easy win .

Try to run TomTom's software under Wine, and you may well get a "device not supported" message. Fail.



HAPPY HACKING

It is nice to show us some coding rules and some languages, but what I really want to learn is how to help out existing projects.

Could the following be done? Review a piece of software, and find a bug or propose a new feature. Go through the particular case for raising the issue (Launchpad, Gitorious, Bugzilla), find the bit of code that needs changing, patch it or propose how to solve it (eg "the for loop does not take into account iteration from zero and causes it to ignore the first item of a list") without actually writing any code. Or maybe correct a translation from German in MikeOS, or go over push, pull, SVN, CVS, Git or Bazaar, or improve documentation. Or take the constructive criticism nicely even if a Linus-school-of-thought lead maintainer calls you a backbirth.

Anyway, please show us how to be outstanding citizens of the free software world!
Andres

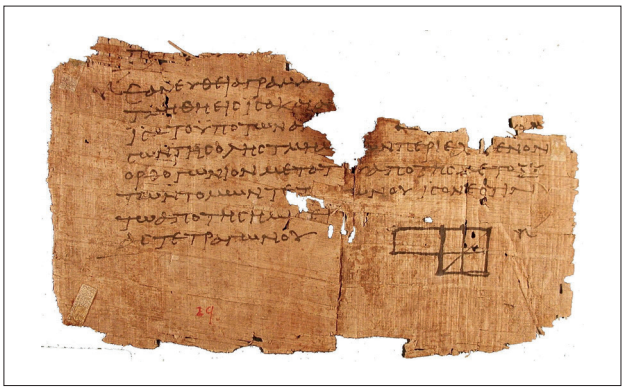
Mike says: This is a good idea. At FOSDEM I spoke to Wolfram Sang, a kernel hacker who gave a talk on improving the kernel without having to contribute any code. I'd like to write an article about this in an upcoming issue. There's also a really niggling LibreOffice bug that I plan to fix – maybe I'll document that. If I don't get totally lost in the huge codebase.



Wolfram Sang wants to make kernel development more accessible – find out more in the next few issues.

MATHS TIME

I was wondering, since as far as I understand this publication is aimed more at the intermediate user level, if you could perhaps do an extensive tutorial series on algorithms. Kind of like a comprehensive guide from the basics to advanced subjects. Algorithms are one of the most important and most fun parts of being able to progam a computer, yet other publications seem to largely ignore them or devote a couple of pages to them.
Tirab



Graham says: Wow – it's like your read our minds! Algorithms are a fascinating subject, so turn to page 100 for a look at Euclid's.

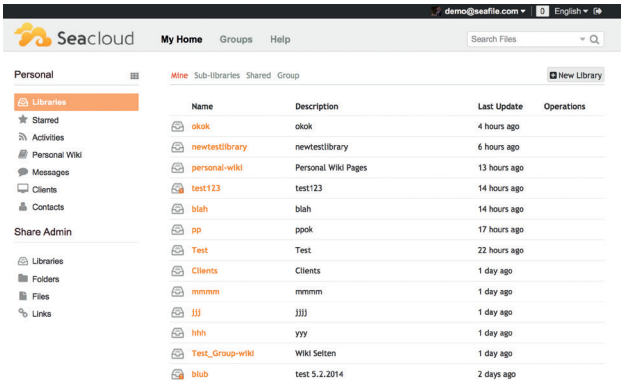
Poor Euclid didn't have access to Emacs, so he had to write his works on scraps of papyrus.

WHAT CLOUD?

I have recently bought a HP Microserver, as my concern for privacy after the Snowden stuff came to light just kept growing. My dilemma now is which self-hosted cloud storage solution to go with. The candidates are SparkleShare, Seafile, Pydio, Syncany or git-annex assistant. I'm struggling to decide and thought you may be able to do a review. It has to be open source and needs to run on Arch.

Ideally I would like to have the files stored in their normal manner, so that I can put the drives in another PC and access them should my server die, ie not encrypted or some special format. Versioning would also be a bonus but not essential.
Andrew Walker

Ben says: Great idea Andrew; we'll try to include it in an upcoming issue.



Seafile is an attractive and open source alternative to DropBox.

OHHH VIENNA

Mike, I wonder what you are doing in Vienna? Are you trying to change the civil service to Linux like it was done in Munich?
Martin Domanski

Mike says: Er, yes, that's right. Honest. Nothing to do with the beer and Schnitzel or anything like that. Incidentally, Graham and I went to Munich to interview the people behind the city's Linux transition, so keep an eye out for our full report next month. (And yes, we did drink Weißbier).



Munich has migrated 15,000 desktops from Windows to Linux – find out how next issue!

LUGS ON TOUR

Blackpool LUG

Our Lancashire correspondent **Les Pounder** reports from Blackpool LUG & Makerspace.

Blackpool LUG meets once a week, making it a rarity, as most groups meet monthly. The venue is the business premises of Mike Hewitt, the LUG's organiser, and just happens to be a PC recycling centre full of old kit that they can use for projects. A typical LUG meet involves brioche and coffee, oh and quite a lot of hacking and making. Past projects include repurposing thin client computers (using Puppy Linux) for Oggcamp signage, and a step-by-step video guide on how to dismantle and rebuild a desktop computer.

Blackpool LUG recently became a makerspace, growing with the group's interests in maker culture and hardware hacking. It now has lots of great facilities, such as a soldering room with extractor fan and data sheets for all of the different microcontrollers that they tinker with. They especially enjoy hacking with Arduinos, and now everyone has an-Arduino compatible device called "The Shrimp", which is a barebones Arduino, but fully compatible with the Uno range of Arduinos.

Every week, there will be three or four different projects on the go. For example, the youngest member, Kieran, has been learning Python to augment his computing lessons at school. His skill has grown each week and now he is readily creating his own programs such as a lottery number generator. Blackpool LUG's most senior member, Elizabeth, has been learning Grub and how to configure it to maximise her workflow, while Mike and Donald, our two electrical experts, have recently repaired an old LCD TV to full working order, enabling the LUG to have a purpose-built Google Hangout device on the main wall.

BLUG on the road

Members of Blackpool LUG have done many great things, both inside and outside of their meetings. They have travelled the UK providing logistical and technical support for many high profile events, you might know of one, the mighty Oggcamp unconference. Blackpool LUG have supported Oggcamp since 2011 and can be found working in the crew along with the other magnificent volunteers.



Photo credit Les Pounder CC-BY-SA

The members working on their individual projects around the communal table notice the coffee pot is empty.

Linux is a community, and what makes a LUG so special are the people, who all have a shared interest in Linux and that bonds us together. But the friendships created are more than this. I'd like to say thanks to Mike Hewitt, for keeping the LUG open over the years and giving us a great home.

If you have yet to find your local UK Linux user group, head over to **http://lug.org.uk** and sign up to your nearest meetings mailing list. Drop them a line and say hi.

If there isn't a LUG in your area, why don't you start your own? For a venue, why not try a coffee shop or local pub? Start small and build upon each success – that's what Blackpool LUG have done.

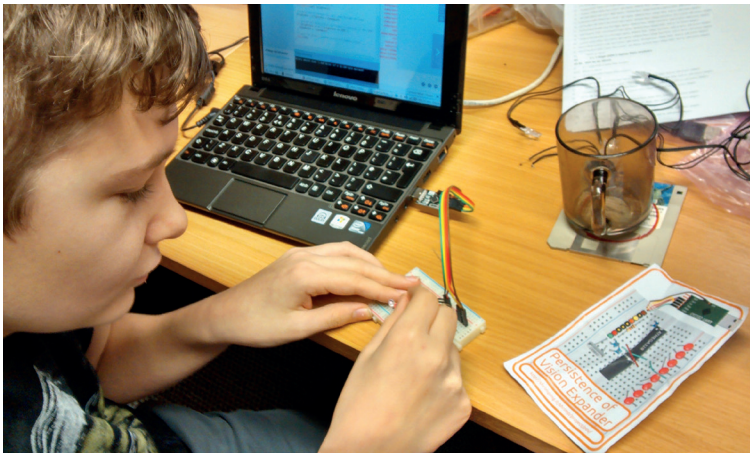


Photo credit Les Pounder CC-BY-SA

Kieran learning more about The Shrimp by building a Persistence of Vision kit.

TELL US ABOUT YOUR LUG!

Chances are that you are already a member of a Linux User Group (LUG). LUGs are all over the world and each one has its own unique selling point, which draws its members to meet and discuss their favourite topic. We want to know more about your LUG or hackspace, so please write to us at **lugs@linuxvoice.com** and we might send one of our roving reporters to your next LUG meeting

Sheffield LUG

Richard Ibbotson on the history of ShefLUG.

"I remember back in the 1990s helping Manchester LUG to come into existence. Dr Owen le Blanc, David Clark, Dr Dave Gilbert, Professor Ted Harding and some other notables used to get together on Saturday afternoons to create the first LUG meeting in the UK. It was at about this time (1992 to 1997) when the people at Manchester Computing Centre were working together with Linus Torvalds that Dr Owen le Blanc created MCC Interim Linux. A Dr Rüdiger Berlich arrived from Germany. I helped him to start SUSE Linux Ltd in London and also helped him with the same thing in California. The SUSE ethic at that time was "SUSE Linux. Have a lot of fun".

Since I am from Sheffield originally I thought that creating ShefLUG was the next logical step. In Sheffield at that time there were a few historians and other academic people who had realised that open source and free software based on the General Public Licence was going to be something more than a music hall joke. Although, they hadn't really understood what I understood. Which was ? That open source software is the future.



Rüdiger Berlich demonstrating SUSE 7.1 in front of Sheffield LUG. Released in 2001, SUSE 7.1 was the first release to feature the Yast package manager/configuration centre.

Gradual improvement

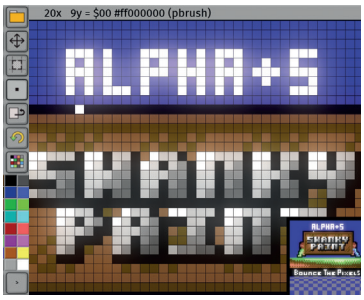
When I first started ShefLUG in 1999 we had a really nasty website. This is much improved in the present day. People didn't really know what they were doing at meetings. We had to slowly evolve a venue for meetings. Eventually we had a solid core of followers. Some of them are still around today. Times have changed. The Android GNU/Linux based operating system and other things like tablets have come along. Recently people at the Linuxcon in Edinburgh were saying "We won": GNU/Linux has taken over the world. Whatever the truth might be I find it best to remember one thing throughout the FUD and fog that is out there. Which is ? GNU/Linux. Have a lot of fun."

Bradford LUG

John McLear and **Shi Hussaini** on a hive of activity.

Shi Hussaini got in touch with us from Bradford LUG to tell us about what they're up to in the land of dark satanic mills:

"My husband and I have been working on Swanky Paint, a Deluxe Paint clone (old school pixel editor used in the games industry since the 80s) targeting the Raspberry Pi natively, as well as Linux, Android, NaCl etc. but the Pi is our minimum spec. So far we have basic editing and animated gif support so it's actually usable!"

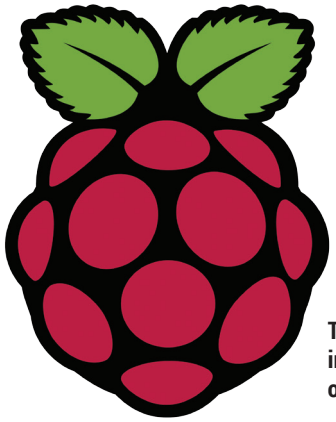


Wallow in the retrostalgia.

One ring to rule them all

John McLear wrote to tell us about a cryotic Raspberry Pi/XBMC project. "I have been hacking on Pi/NXP Explore so I can use an NFC Ring to pause/play VLC/XBMC videos by touching a panel under a TV. Next step is so each person in my house can have a profile GUID stored on their ring and when they touch the TV it plays from a list of tv shows recommended/streamed for them."

We'll admit that we didn't really understand this until we realised the NFC ring is a project that John funded on Kickstarter, raining almost a quarter of a million pounds to develop a ring that can be worn on your finger, and which transmits data to nearby devices.



The Pi is thriving in its homeland of South Wales.

Pi Cymru

Doug Gore reports from South Wales.

Pi Cymru, the Raspberry Pi community for Wales, is hosting their first event - a Raspberry Jam at TechHub in Swansea city centre on Saturday 8 March from 1.30 to 5.30pm. The event brings Raspberry Pi users in the area together to meet in person, learn from each other and find out how they can put their Raspberry Pi to good use. There will be talks, project demonstrations and open discussions on the Raspberry Pi. The event is free and open to everybody including families and children. You can register for the event at **swanseajam.eventbrite.co.uk**, and find out more on our website at **www.picymru.com**.



The crazy cats of The Document Foundation show us all how to run LibreOffice on a Raspberry Pi.

Semi-recursively, the Fedora team used Fedora to print little Fedora badges.

FOSDEM 2014

Free Software, free stickers and delicious beer – **Mike Saunders** and **Andrew Gregory** just had to be there...

Geeks work hard, but they play hard as well. While the vast majority of Linux and Free Software development takes place over the internet, real-life meetups are hugely important as well. One of the biggest such European meetups is FOSDEM, which takes place every February in Brussels. We went along to see the action, and we didn't come away disappointed.

Scattered across various halls of the ULB Solbosch Campus, FOSDEM was packed with talks from developers working on projects across the whole Free Software spectrum: desktop applications, networking, security, games and open hardware. Hackers could watch presentations about new technologies, before exchanging ideas with other programmers and doing some on-the-spot coding.

Many, many speakers

Michael Mrozek introduced the DragonBox Pyra, the follow-up to the Pandora open source handheld games console, while Daniel Naber explained how LanguageTool, a Free Software proofreading program, has been used to find over a million style and grammar errors in the English-language Wikipedia.

The latest build of OpenSUSE works on ARM chips, as shown here on a humble Tegra laptop.

OpenMandriva and Rosa teamed up to talk about porting their distros to ARM devices, and Laurent Eschenauer showed off the Nodecopter, a drone that's programmable with JavaScript.

People power

We met a bunch of awesome people on our travels. Pieter Hintjens, anti-software patents campaigner and author of the ZeroMQ distributed computing framework, kindly gave us a copy of his latest book, *Culture & Empire: Digital Revolution* (see page 28). Red Hat evangelist and "transnational citizen" Jan Wildeboer told us how he disabled the RFID chip in his passport using his microwave – he's a fascinating guy, and we hope to get an interview with him in the next few months.

The canteen, where all the best people hang out and much of the

We also spoke to Wolfram Sang, a kernel hacker who is trying to get non-coders involved in kernel development. How can you improve the kernel if you don't know any C? Well, Wolfram explained how it's fairly easy to add new device IDs to drivers, so if you buy a new webcam and it doesn't work out-of-the-box in Linux, you may be able to add its device ID to an existing driver to make it function. We'll have more from Wolfram in Linux Voice soon, so stay tuned.

Take a stand

But our favourite part of FOSDEM 2014 was the stands. Most computer shows tend to have boring stands with gelled-hair salesdrones parroting the latest blurb about their products, but it's totally different at FOSDEM. Small projects are well represented, and the people at the stands are the geeks who work on them. Anyone could walk up and chat to developers of the Enlightenment window manager, for example, and the same goes for CentOS,

automated distro testing system (that sends keyboard and mouse input to the VM window to perform automatic installations), while the Fedora stand wooed everyone with an awesome 3D printer creating Fedora badges. Also present were the Free Software Foundation Europe, the Mozilla Foundation,

Beer was tasty and plentiful; the WiFi network did a good job as well, given that there were several thousand geeks connected to it at any one time. It was great to meet so many Linux Voice fans as well – your support and enthusiasm is amazing. So, thanks to everyone who attended and showed off their awesome work, and see you next year! LV

The relaxed crowds and fast-food vans give FOSDEM the feel of a music festival rather than a tech conference.

LINUX VOICE

REVIEWS

The latest software and hardware for your Linux box, reviewed and rated by the most experienced writers in the business



Andrew Gregory
When he's not gently massaging his Linux Mint installation, Andrew enjoys a game of Oolite.

Some things change without you noticing them. For years we've seen LibreOffice come on in leaps and bounds, adding stacks of new features and better support for non-native file formats. We've pored over every update and marvelled at how the suite has become a challenger to – and in many ways the superior of – Microsoft Office.

What hasn't been so immediately evident is the vastly improved performance. I only thought about this the other day: LibreOffice starts in a couple of seconds now. OK, so I have a fairly modern PC with an SSD drive, but I thought back to the bad old days of OpenOffice.org 1.x and 2.x, which would take around 15 seconds to start.

A huge amount of work has been done under the hood of LibreOffice to enhance its performance. Our good chum Michael Meeks has been responsible for much of this work, and kudos to him: it's not a glamorous job, but someone has to dig into the guts of the codebase and remove the cruft.

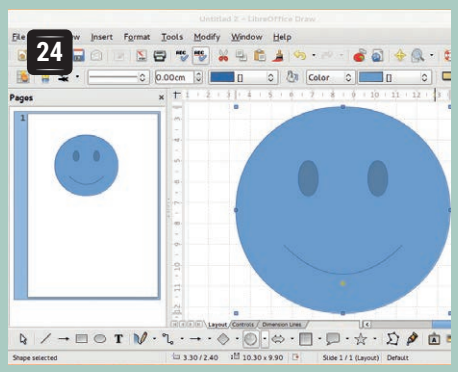
andrew@linuxvoice.com

On test this issue...



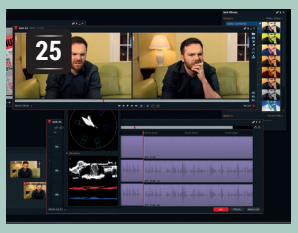
Mageia 4

We still can't agree on how to pronounce its name, but this Mandriva spin-off has lasted longer than some thought it might.



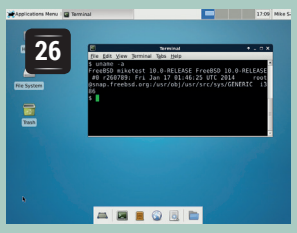
LibreOffice 4.2

Ben dives head first into the latest release of this mighty office suite – are the days finally numbered for Microsoft?



Lightworks 11.5

It ain't cheap, but this pro-level video editor is brimming with features.



FreeBSD 10

Unix goodness with a more conservative development model. And some daft bugs.



Nexus 5

Fancy hardware and no silly OEM bloatware: is this the perfect smartphone?

BOOKS AND ROUNDUP

Trees are good. They release oxygen into the atmosphere and prevent us from dying of asphyxiation, and they can also be turned into thin slices containing written material. This month we look at *Culture and Empire: Digital Revolution*, a great read explaining how the powers-that-be are waging war on the internet.

Over in the Group Test we look at encryption software: given the amount of spying going on, it's the right time to start encrypting your communications. We look at the best tools for the job.



Mageia 4

Graham Morrison takes on the 64-bit version of one of the strongest distributions to appear in recent years. Vive la différence!

DATA

Web
www.mageia.org
Developer
The Mageia Community
Price
Mostly Free Software

We dove straight into a fully fledged installation off the 64-bit DVD, rather than playing around with the live CD first. As ever with Mageia, the installation process looks fantastic. It recognised our drives and made it clear which drive we'd selected and what the consequences of installing might be. We used the manual partitioning option and created a new root partition from free space on our drive, although clicking on Done quickly brought up the formatting dialog without any final chance to check the settings – a security blanket we'd got used to with Ubuntu.

We were also able to choose between a vast array of desktops, which is a defining feature of Mageia 4. KDE and Gnome were the main options, but clicking on 'Custom' brought us to the 'Package Group Selection' screen, familiar to any former Mandrake user. This page gives you a great overview of which packages are going to be installed, enabling you to freely change what you'd like. From here, for instance, you can install Xfce, Mate, Cinnamon, RazorQt, E17 and LXDE without resorting to the package manager after boot. We've never seen such a comprehensive selection of desktops from a single DVD, and we clicked on them all – which is perhaps why it took 80 minutes and 5.5GB to install from the DVD. After package installation had completed, we did have problems telling the installer not to install a bootloader – see the box on EFI And GPT Partitions – but otherwise, this was a first-class process.

Boot camp

Our first boot was delayed for a considerable time in the Checking For New Hardware state, but this didn't happen with second and third boots, so we're guessing this had something to do with the many USB devices we've got connected to our system. Logging into the default KDE desktop bought back many happy memories of Mandriva, mainly because Mageia uses a similar blue colour scheme. But it's also that Mageia is augmented by many of the same tools that made Mandriva so good. The Mageia Control Center, for example, still has its own icon in the toolbar and provides easy access to software installation, hardware configuration, networking and system management, in exactly the same way DrakConf used to in the olden days.

We also really like the MageiaWelcome application, which presents itself when you first reach the desktop. This is a great tool for beginners, featuring

Mageia defaults to a great KDE desktop, but also offers the most unrivalled selection of alternatives we've ever seen in a distribution.



direct links to release notes, errata, how-tos, the wiki, forums and chat rooms. Unfortunately, the last link will only take you as far as launching X-Chat, leaving the beginner to discover for themselves how IRC works and where to find Mageia's support group. Not a great feature.

KDE 4.11 is the version shipped with Mageia 4. The launch-menu is the classic 'click and execute' style, as you'd find on Windows 98, but we miss the pervasive search of the application launcher – enabled with a right-click whilst the widgets are unlocked. Sound works perfectly through the GStreamer back-end to PulseAudio, and it detected our complex array of kernel audio devices without difficulty.

We wish a KDE distribution would be brave enough to drop KDE's default blue 'Active Window Glow', inherited from the Oxygen window decorator, but this can also be changed when you know where to look. There's a great selection of the best KDE applications including Amarok, Digikam, Kopete and KMail, but its office suite is deprecated in favour of LibreOffice.

Desktopification

The available software from a default installation is fantastic, making Mageia a great one-click install option for a comprehensive Linux/Open Source software suite. Having tools like Audacity, FileZilla, Ekiga, Scribus and Abiword just a click away is great for office environments, or for friends and family, although we'd like to see VNC added to the mix. Our installation had Kdenlive running in the background, for some reason, taking up 99% of a single CPU core. We then took a quick tour through the various installed desktops; it's great that you're able to do this after only just installing the system from a DVD.

Cinnamon was nicely accelerated, although the bottom panel was a little small on our high-resolution screen. Enlightenment 17 was its minimalist self, although asking us for our preferred size of window title bar was a nice touch. We'd have to question Inkscape's high-priority placing in the toolbar, but Enlightenment on Mageia fels fantastic – slick, polished and lightweight.

Next up in the login menu is Gnome 3.x, which we have to admit is starting to make sense after years in the wilderness. Apart from the background and the welcome window, this is the default Gnome experience. IceWM follows Gnome in the desktop selection menu, and this gives perhaps the most basic desktop experience of the lot. The background image isn't scaled, for example, and there are no icons in the launch menu. But that's IceWM, and it's good to have it here for low-powered Linux machines. We prefer LXDE for the lightweight experience because it has just a few more niceties, such as icons, but they both work well in Mageia. Mate, too, is a fuss-free two panel desktop. It's quick and will feel familiar to anyone used to an older version of Windows, and looks great with reasonable sized anti-aliased fonts and panels. Openbox wins for absolute minimalism, so we'll just say it works. More interesting was RazorQt, and this is the first time we've tried this nascent Qt desktop from a default installation. Perhaps because we had so many others installed, we could choose between four different window managers. The Razor desktop became our favourite low-resource option simply because KDE apps felt almost native in this Qt environment. Finally, there's Xfce 4, which we also love, because of its great default options.

This amazing selection of desktops available from a single installation DVD is Mageia's killer feature, and serves a great purpose for people who may be finding their way with Linux desktops. And this is definitely a distribution we'll keep around for updates, because it bundles all these desktops together. But Mageia 4 is also a serious distribution, full of tier-one free software and a modern emphasis on kernel updates and security – Shorewall is running, for example. It's also one of the few distributions where suspend and resume has worked. With a few caveats about EFI installation and custom control panels, Mageia 4 is highly recommended.

> **"The amazing selection of desktops available from a single DVD is Mageia's killer feature"**

LINUX VOICE VERDICT

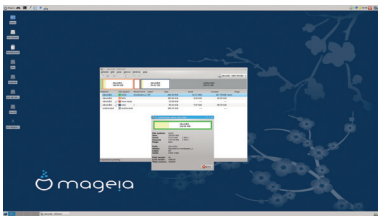
Despite a few rough edges, this has to be the distribution to choose if you're a serial desktop swapper.

★★★★½

EFI AND GPT PARTITIONS

It's a little unfair to pick on Mageia for not working with GPT and EFI, because this is a problem that's only just beginning to emerge for many distributions. But its release notes do boast 'experimental support for UEFI', so we're going to mention it. GPT and EFI replace the old partitioning and boot schemes that we've used for 20 years, and it's only now that PCs are appearing on the market that default to EFI rather than a legacy BIOS option. With Mageia, one of the biggest problems we found installing on one of these systems was that you couldn't choose to ignore the bootloader. The only options were to install Grub, Grub 2 or even Lilo! We wanted to skip this stage so we could sort out our EFI bootloader ourselves, but in the end, the only way forward was to install Grub 2 onto the partition itself, rather than the non-existent MBR.

After installation, we were able to get Mageia booting off GPT/EFI without too much difficulty. Using gummiboot, a very simple bootloader, we copied the kernel and initramfs images from Mageia's **/boot** folder into our EFI partition, and created a simple configuration file for gummiboot that pointed to the location of both, as well as set the root device for our installation. As we were using Nvidia's proprietary drivers, we also had to include **nokmsboot** as a boot option within the configuration file too. But after this was set, booting was perfect and we had no further issues. We'll just have to keep an eye on any kernel updates so we can copy the files over again if necessary.



EFI support in modern distributions is becoming increasingly important

LibreOffice 4.2

It's not often an office suite excites us, but **Ben Everard** discovers LibreOffice 4.2 has a trick up its sleeve.

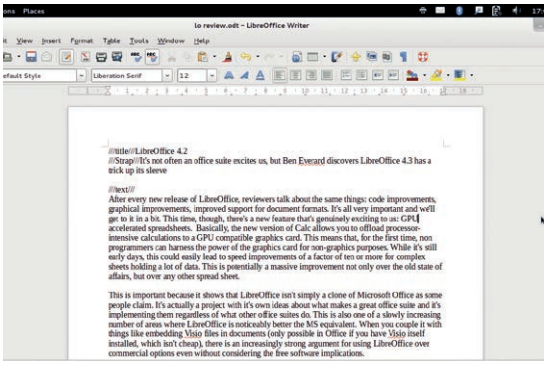
DATA

Web
www.libreoffice.org
Developer
The Document Foundation
Price
Free under LGPL

After every new release of LibreOffice, reviewers talk about the same handful of things: code improvements, graphical improvements, improved support for document formats. It's all very important and we'll get to it in a bit. This time, though, there's a new feature that's genuinely exciting to us: GPU-accelerated spreadsheets. In a nutshell, the new version of Calc enables users to offload processor-intensive calculations performed in the spreadsheet to a GPU-compatible graphics card.

This means that, for the first time, non-programmers can harness the power of the graphics card for non-graphics purposes. While it's still early days, this could easily lead to speed improvements of a factor of ten or more for complex sheets holding a lot of data. This is potentially a massive improvement not only over the old state of affairs, but over any other spreadsheet out there.

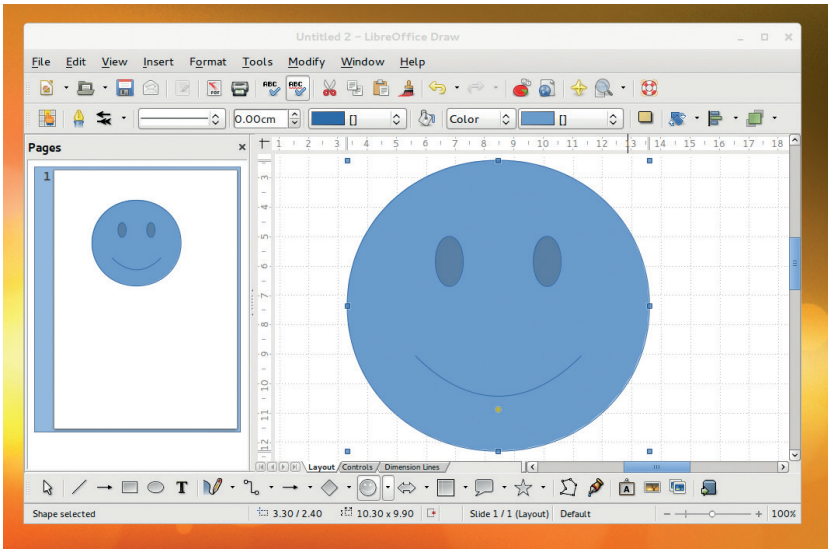
This is important because it shows that LibreOffice isn't simply a clone of Microsoft Office, as some people claim. It's actually a project with its own ideas about what makes a great office suite, and it's implementing them regardless of what other office suites do. This is also one of a slowly increasing number of areas where LibreOffice is noticeably better than the Microsoft equivalent. When you couple it with things like embedding Visio files in documents (only possible in Office if you have Visio itself installed, which isn't cheap), there is an increasingly strong argument for using LibreOffice over commercial options even without considering the implications of free software.



LibreOffice Writer one of the key programs we use to create *Linux Voice*. Thanks, Document Foundation!

Not only is the GPU-accelerated spreadsheet a great new feature, but it also shows the benefits of open source code. It was developed in part by AMD, which could only do this because it had open access to the code.

> "LibreOffice isn't simply a clone of Microsoft Office, as some people claim."



The smiley in LibreOffice 4.2 (shown here in Draw) has been improved to make it happier.

Free as in better

There have apparently been improvements to the import of MS Office files, though we're not sure where, as it's been a long time since we've had any problems importing files. Code re-factoring and translation of German comments to English are both featured so much in LibreOffice press releases that they've become clichés, but they are of course important. Once done (it'll take a few more releases), this should make it far easier for developers to add new features (many more developers speak English than speak German), so although it doesn't sound exciting now, it's preparing the code for exciting things in the future.

The crux of any review is the question: should you install this software? Unless you're having issues with the old version, or have a large spreadsheet you want to offload to your GPU, there's probably little to be gained from rushing out and grabbing this version rather than waiting for it to appear in your package manager. What's exciting is the way that, with this release, LibreOffice has taken such a commanding position in the office suite market.

LINUX VOICE VERDICT

Another solid release of LibreOffice with a great new feature – exactly what we've come to expect.

★★★★½

Lightworks Pro 11.5

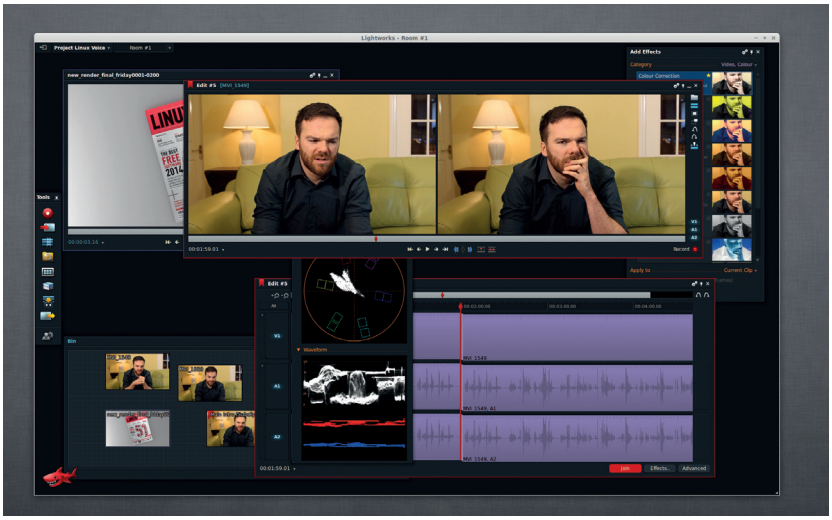
It's taken a long time and a metric ton of development work to get here, but has **Nick Veitch** found the Final Cut Pro for Linux?

Over recent months, we've learnt quite a bit about video editing. From lighting, sound and taking multiple shots, through to getting the best from the software we use. And unfortunately, one of the toughest lessons was that there just wasn't a good video editor for Linux. Lightworks could be our saviour. It's been around on Windows for a long time, and a Linux version was announced way back in 2012 – alongside a commitment to release an open source version, which has yet to materialise. After a beta release last year, 11.5 is the first stable version for our favourite operating system, and is something of a milestone for non-linear video editing on Linux.

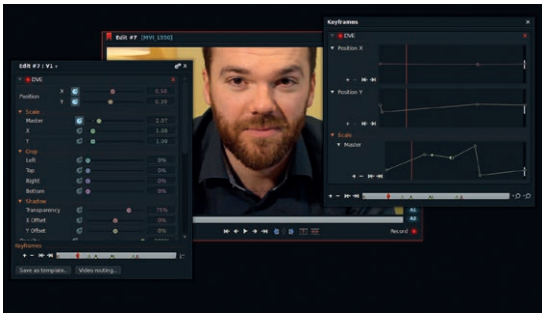
Lightworks has a free version and a Pro version. The free version features the same editing environment but is restricted to web-friendly output formats and a 720p output resolution. The Pro version unlocks broadcast output formats and resolutions, and even 4K and 5K as image sequences. Ubuntu and Fedora derivatives are officially supported, and we installed the 60MB download on 64-bit Mint 16 by simply clicking on it. The recommended computer specification is on the demanding side (see side panel), but that's not surprising. Video editing anything other that cat videos from your smartphone requires as much CPU and memory as you can throw at it, with your GPU in particular used to accelerate many processing tasks.

The Final Countdown

Coming from Final Cut Pro (Apple's market leader), Lightworks takes some getting used to, although many of the same features and facilities are present. Lightworks gives you more freedom to organise clips, create, composite and pull these together into edits, which themselves can be stacked and re-used in a very modular way. Clips can be dragged against one another, changing the time in both, or moved around on a timeline. Selecting Delete will split a clip, and you can drag and drop other clips, transitions, audio and images. We missed embedded thumbnails, but they're not essential, and it's not always obvious where features hide. The all-important wipes and transitions, for blending from one shot to another, are hidden behind a press of the F8 button, for example, with no menu or icon we could see to indicate its existence. The selection of effects is brilliant, and give the editor a fantastic palette of powerful processing that remains functional rather than over-stylised, which is the problem with many effects in Final Cut Pro. Our favourite feature is the all-powerful keyframes, which can easily be used to change parameters over time, creating slow-pans or focus effects with aplomb. Keyframes can even be edited from a graph, and while we couldn't find an option for Bézier curves, there's more then enough detail here to get exactly the results you need.



Lightworks defaults to using your entire screen, but it can be coaxed into a window.



Keyframes are brilliant for changing parameter values over time. Look at that beautiful face.

There's nothing like Lightworks on Linux. It's a professional video editing suite that really does deliver results. If you make money from editing, you won't begrudge the cost. If you don't, the free version is good enough for many projects – with the 'Creative Cloud' like pricing giving a reasonable entry point for one-off projects.

> **"Lightworks is a professional video editing suite that really does deliver results."**

DATA

Distros Ubuntu 13.10, Mint 15, 16, Fedora 18, 19
Intel i7 or AMD equivalent
3GB RAM or higher
1GB PCI-Express VGA **NVIDIA** or ATI with OpenGL
Web www.lwks.com
Developer EditShare
Price £180 or £49.99pa

LINUX VOICE VERDICT

It's the only application of its calibre on Linux, and in terms of features, we've barely scratched the surface.

★★★★★

FreeBSD 10.0

Without great fanfare, the FreeBSD project keeps pumping out releases. **Mike Saunders** explores what's new in version 10.

DATA

Web
www.freebsd.org

Developer
The FreeBSD Project

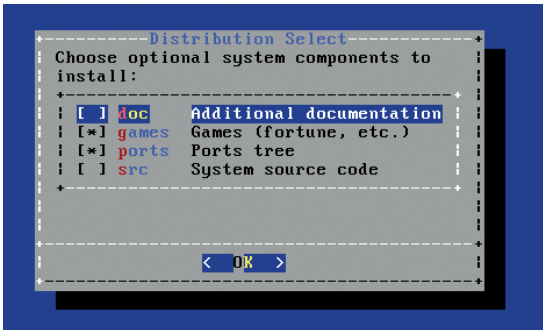
Price
Free under the BSD license

For many of its users FreeBSD is a more traditional operating system than Linux: it's Unix-like, it's free, it's open source, it's reliable, and it carefully integrates new features in a gradual development cycle. The entire FreeBSD source code tree, including the kernel, core libraries and system utilities, is developed as a whole – in contrast to Linux, where one group maintains the kernel, another maintains glibc, and so forth.

FreeBSD 10 is available in various formats, from traditional CD/DVD ISOs to a **.img** file that can be written to a USB key. The installer is largely unchanged from 9.x; it's a text-mode tool reminiscent of older Linux distro installers, and while it's not pretty it gets the job done quickly, and is much simpler than the plain text OpenBSD installer.

In general use, FreeBSD looks and feels much like a GNU/Linux distribution. Almost all major open source programs have been ported to FreeBSD (often requiring no modification), and there's a fairly reliable Linux compatibility layer to run Linux-only binary programs. As a desktop OS it's close to Linux in features and performance – although hardware support isn't quite as extensive. Under the hood many command options and filesystem locations are different, and the OS expects more prior knowledge than a newbie-oriented Linux distro, but the great documentation makes things clear.

Most notably, GCC is gone from FreeBSD 10's base system, being replaced with LLVM/Clang. The FreeBSD team has wanted to switch to a BSD-licensed compiler for years, and now it has. (GCC is still available to install, though.) ARM support has been extensively improved, so FreeBSD now runs on the Raspberry Pi, while a new type-2 hypervisor (bhyve) has been added along with support for running on Microsoft's Hyper-V.



The installer, bsdinstall, was originally created as a "stop gap" tool but has stuck around since 2011.

TRIM and LZ4 compression has been added to the ZFS filesystem, and FUSE (filesystem in userspace) is now in the base system, so you can use all of the awesome FUSE drivers that were developed on Linux. Then there's USB audio 2.0 support, wireless networking improvements, and boot time speedups. It's an impressive bunch of enhancements touching almost every area of the OS.

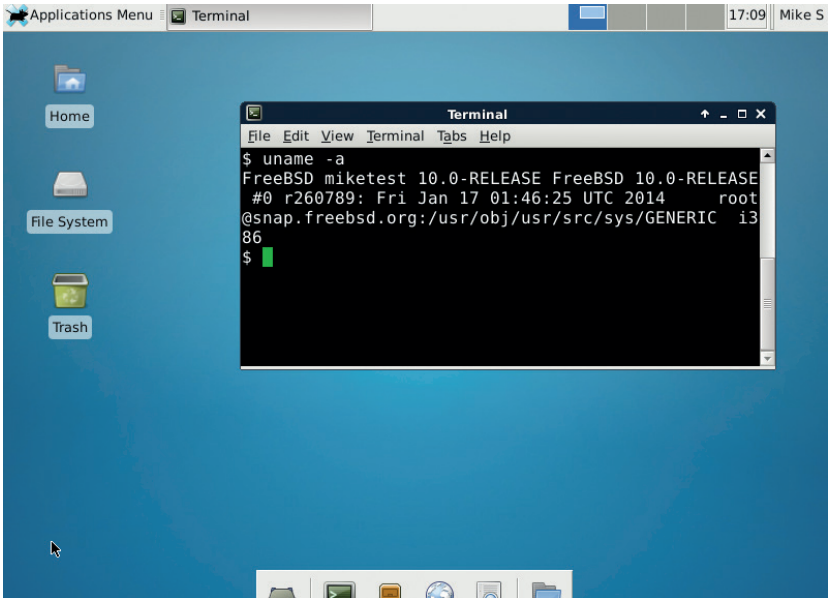
Regrettable slip-ups

A major setback for FreeBSD 10, though, is the embarrassing bugs that have somehow slipped into the final release. Sure, major new versions of software are going to have some flaws, but the issues in FreeBSD 10 should have been caught much earlier. Look at **www.freebsd.org/releases/10.0R/errata.html** and you'll see alarming examples like this:

"A bug in killall(1) has been discovered. It makes killall -INT deliver SIGTERM rather than the desired SIGINT, and may cause blocking behaviour for scripts that uses it, as -I means 'interactive'."

This is very alarming, and the same can be said of the pw(8) bug, which changes the behaviour of a critical system tool. When an OS prides itself on stability and a conservative development process, for such silly bugs to slip through is awkward. It doesn't mean that FreeBSD 10 is a disaster – far from it – but it's bad for the project's image.

FreeBSD's phenomenally stable base system doesn't include anything graphical, but a quick 'pkg install xorg xfce' command will get you a usable desktop.



LINUX VOICE VERDICT

Plenty of new features to explore, but the silly bugs let it down. Here's hoping it's just a one-off.

★★★½☆

Google Nexus 5: three-month review

After some time with Google's flagship smartphone, **Graham Morrison** reaches a conclusion on whether it's worth the upgrade.

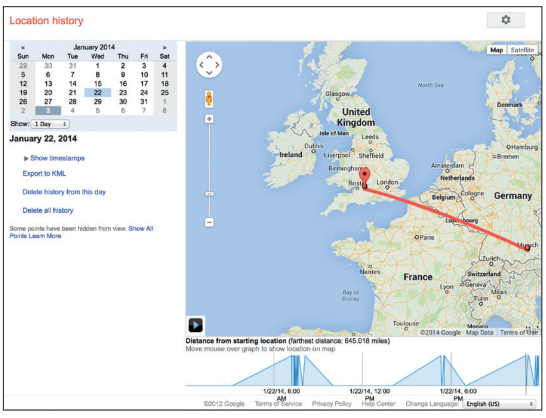
A couple of us on the team have a Nexus 5. On paper, when value is a consideration, it's the best phone you can buy, and will likely stay as Google's flagship device for some time yet. But price/specification comparisons don't tell you the whole story. You have to spend time with a device to be able to say whether it's really a successful product. Which is exactly what we've done, reporting our findings on whether the Nexus 5 is still the go-to phone for Linux-loving smartphone users.

After a flaky start, Android has become a wonderful operating system. When notifications appear in the top-left of the display, they're subtle, functional and easily swiped away. With a few choice applications, such as 'Status Agenda' for adding appointments to the notification area, a replacement launcher or a better text messaging application app than the default (the awful Google Hangouts) you can build an environment that works best for you, which is something that can't be said about Apple's iOS.

Hardware

The hardware is fantastic. After three months' daily use without a case, the screen is still unblemished. Nor have we noticed any problems being understood through the diminutive microphone grille – which some users have complained about.

Speaker output is quite low for hands-free, and the quality of audio recording is about only average – it's legible, but it's not crystal clear. The headphone outputs are also on the low side, although subjective audio quality is excellent. And the camera was vastly improved by an Android update. The screen, as it was in the beginning, is stunning, but the auto-brightness setting can be hit and miss. With average use, our phones last us about 40 hours without a charge, which is perfectly usable.



If you want to scare yourself, take a look at the location history for your account. Google is watching you.



Even though the screen on the Nexus 5 is larger, the phone feels thinner and lighter than the Nexus 4.

DATA

Web
www.google.co.uk/nexus/5
Manufacturer
LG
Price
£299 for 16GB version; £339 for 32GB

We do have problems with some aspects of Android, and by far our biggest is with its unrelenting appetite for personal data. Google Now begs you to turn on browsing history, and refuses to unlock its best features until you do. Every time you open your photos you're pleaded with to enable online backup. Many things seems enabled until you turn them off, and you only have to look at your Google location history to get a terrifying snapshot of where you've physically been, helpfully logged by your phone even if it's just down the road to get some milk.

We're uncomfortable with this, which is why Cyanagonmod may be our best option. Cyanogenmod is Android with the Google cloud services taken out. Similarly, the F-Droid repository seems to be going from strength to strength, and it's become an essential resource for open source apps and utilities. The OwnCloud app, for example, automatically uploads photos to your server, and is particularly helpful, as are the terminal and SSH clients. We still don't like SMS and chat integration, but a small SMS icon now tells you whether you're chatting on your mobile network. Fortunately, you can download the previous version of Messaging through Google Play, thanks to again to Cyanogenmod.

> **"Our biggest problem with Android is its unrelenting appetite for personal data."**

Not withstanding privacy issues, the ability to customise nearly every aspect of your phone's operating system will appeal greatly to the average Linux user. And we can't argue with that.

LINUX VOICE VERDICT

A wonderful phone made even more wonderful by the availability of Cyanogenmod.

★★★★★

Real World OCaml

Mike Saunders starts learning his eighth programming language.

O'Reilly doesn't shy away from covering esoteric topics. Few developers have ever dabbled in OCaml, but this object-oriented, functional programming language has a lengthy history behind it, extending Caml (which itself is a dialect of ML, stretching back to the early 70s).

This book describes OCaml as "an industrial-strength programming language designed for expressiveness, safety and speed", and the opening chapters do a good job of setting out the advantages of using the language.

Weirdly, *Real World OCaml* avoids using the language's standard library, instead basing its teaching and examples on the Jane Street Core library, an alternative with more features and syntax extensions. Fair enough – but this Core library isn't yet supported on Windows. Instead, the book recommends that Windows users install Linux in a virtual machine. This isn't a big deal for us Linux users, but if you're planning to learn OCaml for doing some cross-platform coding, it's worth bearing in mind.

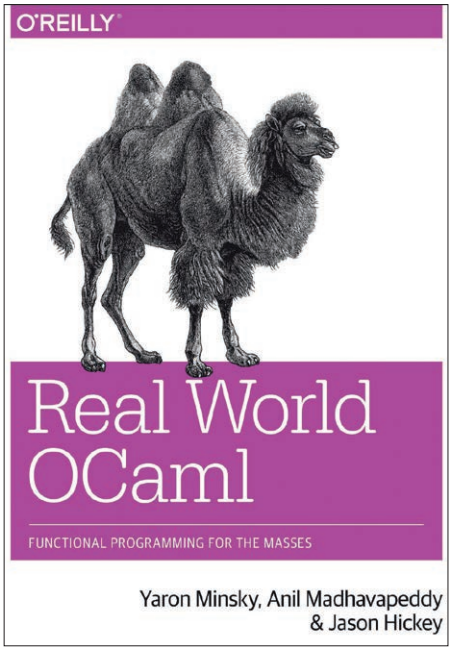
Otherwise, the book is typical O'Reilly fare: 483 pages of minimal presentation and serious text, but it's extremely clear, well written and authoritative. There's no hand-holding here, so if you're completely new to programming you might find the pace too fast. But if you're already well versed in a couple of languages and want to get into OCaml, this is a good choice.

LINUX VOICE VERDICT

Authors Yaron Minsky, Anil Madhavapeddy and Jason Hickey
Publisher O'Reilly
ISBN 978-1-449-32391-2
Price US $39.99

Typical O'Reilly: a no-nonsense, extensive and confidently written guide to all aspects of the language.

★★★★☆

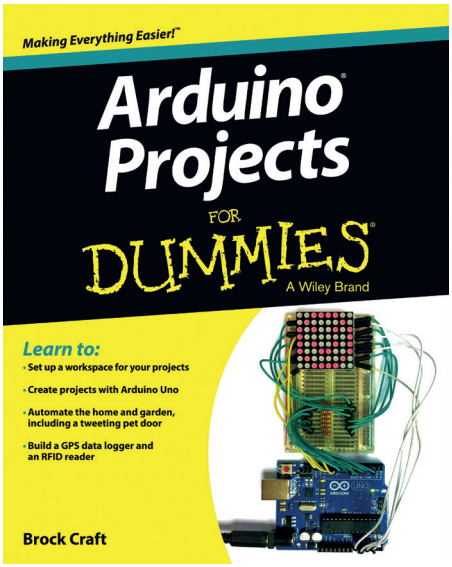


It had to be a camel, of course; but we wonder how Perl will feel about having to share?

Arduino Projects for Dummies

We're not sure how many dummies use Arduino, so **Ben Everard** read it instead.

Arduino Projects for Dummies starts with the assumption that the reader knows very little about either electronics or the Arduino, and the first section introduces both of these subjects. The writing's clear and easy to follow as you might expect from an experienced teacher like Brock Craft.



The focus of the book, though, isn't these introductory explanations, but a series of 12 projects that get harder from simple LED controllers to a remote-controlled car and GPS data loggers. The projects are all achievable for someone with limited experience and time. After getting the basic skills from the basic projects, most people should be able to take on even the more advanced projects. The clear writing and clear diagrams make them easy to follow. Because of the ecosystem of Arduino shields and libraries, features like RFID tags and accessing Twitter are easily achievable and this book introduces them in a fun way.

The book does cover building projects well, and looks at a number of common shields and simple components. However, it completely passes by some things that you'd need if you wish to take things further by yourself. For example, I2C gets just one paragraph, but this is still better than SPI, which merits only a single sentence.

It would be unfair to heavily criticise the book for these omissions though, because it doesn't claim to be an all-encompassing guide to the platform, but a projects book to help the reader quickly and easily get started with fun builds. It does this admirably. In this context, then, the focus on just what you need to know, rather than getting bogged down in excessive detail, is a point in its favour rather than a downside.

LINUX VOICE VERDICT

Author Brock Craft
Publisher John Wiley & Sons
ISBN 978-1118551479
Price £17.99

Some useful projects to help you get started with the Arduino Uno.

★★★★☆

Culture & Empire: Digital Revolution

Andrew Gregory gets out his tin foil hat, then puts it away again.

Culture & Empire: Digital Revolution, by Pieter Hintjens, is a remarkably clear-sighted overview of how mass connectivity leads to social change and provides an examination of why some forces are resisting this change

Hintjens' analysis is part sociology, part free-market economics, part Marxist theory, but it coalesces into a convincing argument (admittedly with the odd digression). If you've looked around you at state surveillance, SOPA and David Cameron's Porn Filter™ with growing unease, this book is unlikely to make you feel more comfortable; instead it clarifies exactly why your gut feeling is right and helps intellectualise the reasons why the powers that be want to control us.

If that sounds abstract and academic, it isn't. For a work of social theory there's very little of the made-up nonsense words that usually pervade the subject; instead, the writing is clear, explaining complex ideas in simple terms that make you think without having to work on your vocabulary. This book is the serious work that this serious subject deserves.



Culture & Empire is licensed CC-BY-SA-3.0, so the author is practising what he preaches.

LINUX VOICE VERDICT

Author Pieter Hintjens
Publisher iMatix Global Services
ISBN 978-1492999775
Price £14.93 print
£1.90 (Kindle)
PDF download Free

Goes on occasional flights of fancy, but retains throughout a core of truth. Highly recommended.

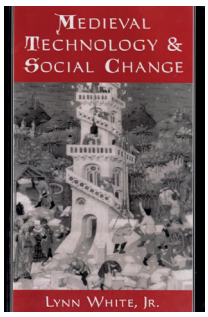
★★★★½

Mediaeval technology & social change

Can **Graham Morrison** learn anything from Mediaeval tech?

This book was recommended to us by Robert 'r0ml' Lefkowitz when we spoke to him last year. It was published in 1962, but r0ml had us enraptured by its principle argument – that the appearance of the horse stirrup in medieval Europe was a precursor to feudalism. The book's thesis explains that this form of combat, where a knight wields heavy weapons while remaining in full control of his steed, altered the structure of society because it was such an effective form of combat. Those knights were bribed into service by offers of land which, in turn, was farmed by peasants pledged to serve their master in battle.

r0ml was working on a theory that used some of these ideas, replacing the stirrup with the computer and pondering feudal emancipation. It's all about resources, and as long as open source exists and everyone is taught how to code, we'll be free from bondage forever – huzzah! Or something like that. Later chapters aren't quite so interesting, unless you're interested in the northward shift in European powerplay. Regardless, it's a fascinating book that's easier to read than you might imagine for an old title devoted to constructing an academic thesis.



What can we learn from the tech shifts of 1,200 years ago? Quite a bit, actually…

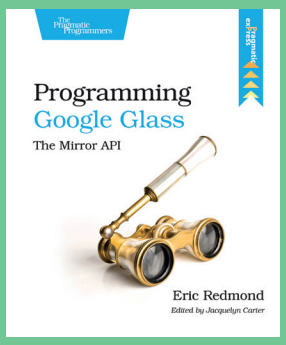
LINUX VOICE VERDICT

Author Lynn White, Jr
Publisher Oxford University Press
ISBN 978-0195002669
Price £13

The perfect excuse to raid every second-hand bookshop you see.

★★★★☆

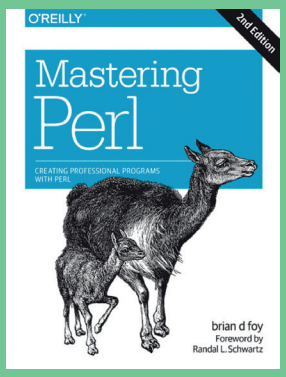
ALSO RELEASED…



Get ahead of the curve by coding Glass apps today.

Programming Google Glass

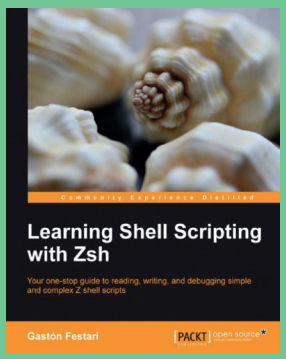
Even if Bruce Sterling isn't a fan, there's no doubt that Google's wearable tech has already had a huge impact. You could probably use this book to write your own inane bird physics game and take over the world.



perl -n -e 'tr/[a-z]/[A-Z]/;print'

Mastering Perl, 2nd Edition

The definitive tome on Perl programming gets an upgrade. This is not light reading for the bath. It's not even heavy reading. It's about Perl, and so exists on its own plane.



The cool kids don't use Bash. They use Zsh.

Learning Shell Scripting with Zsh

Zsh is awesome – you don't know it yet. It can do so many cool things that Bash can't, and also makes you look more l337 at conferences and LUG meetings. This book should tell you everything you need to know.

LINUX VOICE

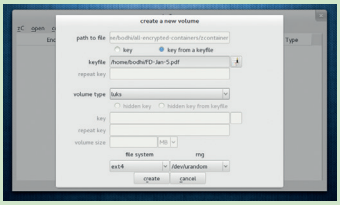
ENCRYPTION

GROUP TEST

Mayank Sharma gathers the best file encryption tools to help you take charge of your personal privacy.

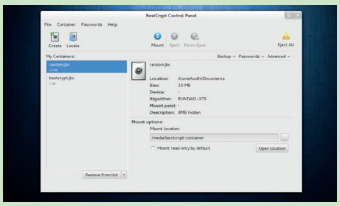
On Test

TrueCrypt



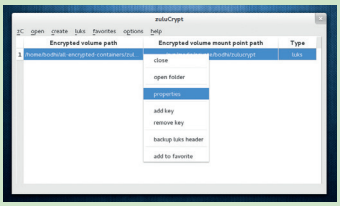
URL www.truecrypt.org
Version 7.1a
Licence
The TrueCrypt licence
Multi-platform graphical solution.

BestCrypt



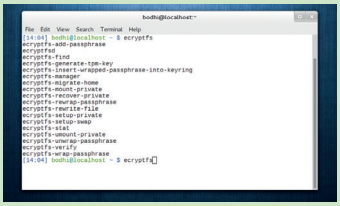
URL www.jetico.com
Version 2.0-3
Licence Proprietary
The paid-for solution – does it offer enough to justify the price tag?

zuluCrypt



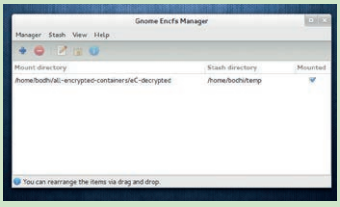
URL http://code.google.com/p/zulucrypt
Version 4.4.7
Licence GNU GPLv2
Flexible, graphical and intuitive encription.

eCryptfs



URL http://ecryptfs.org
Version 103
Licence GNU GPLv2
Speedy encryption that works at kernel level.

EncFS



URL www.arg0.net/encfs
Version 1.7.4
Licence GNU GPLv2
Command-line utility powered by FUSE (filesystem in userspace).

Encryption

Let's define what we mean by this vague term.

The Snowden revelations have reinvigorated interest in personal privacy. You can control access to the data in your computer with the use of file permissions and user accounts. But this type of protection isn't enough to deter a determined intruder. The only sure way to keep your personal data to yourself is to encrypt it. Working with encrypted data is an involved process, but it'll go a long way in reinforcing your security and insulating your data from unwanted attention.

You'll find lots of open source encryption software in your distribution's package repository. Some leading distributions, like Fedora, Ubuntu and Linux Mint, even let you encrypt your entire disk while you are setting it up.

Then there are applications that will help you create encrypted silos within your filesystem, or entire partitions. The hallmark of these applications is that they can do on-the-fly encryption. This means they will automatically encrypt your data before writing it to the disk and decrypt it when called for, assuming you have the right credentials.

On the downside, there's usually a performance hit with transparent encryption, depending on the type of cipher you've used to scramble the data. However some tools can take advantage of dedicated cryptographic hardware extensions built into many modern desktop processors to minimise the performance degradation.

Now then: which of our chosen contenders is best for you?

> **"You'll find lots of encryption software in your distribution's package repository"**

THE CRUCIAL CRITERIA

All the tools were installed using their recommended installation mechanisms. Tools that were easier to install and use were rated higher. However, encryption is a complicated subject that needs to be handled with care, which is why we also looked at the help and documentation offered.

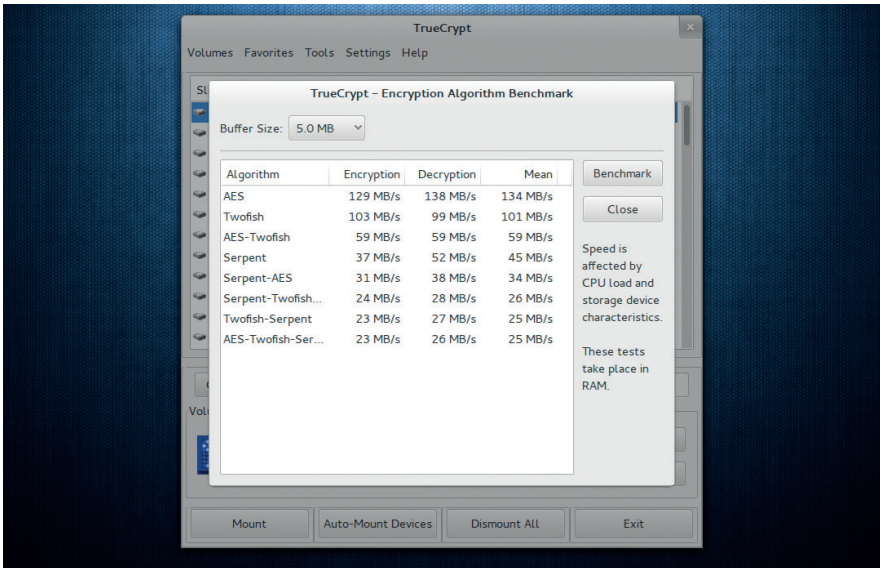
We didn't test the security provided by the tools, because they all use industry-standard ciphers to encrypt the data. We do however take note of the ones that enable their users to select the encryption cipher and its strength.

We also look at usability with respect to their feature set: a feature-rich command-line app isn't necessarily a better option than a simpler but intuitive graphical tool that gets the job done.

Finally we also test the performance of each app by creating containers of equal size using the default ciphers. We then copy a bunch of small and large files and time the operation with the time command. This isn't by any means a reliable benchmarking technique, but it should give you an idea about the relative performance of each tool.

Supported ciphers

Do they meet industry standards?



Attackers will try to break your password rather than the encryption, so make sure you use a long and random password with special characters to mitigate dictionary attacks.

All encryption software relies on a cipher to encrypt. A cipher is an algorithm that does the encryption and decryption. Also important is the key size of the cipher that's used to encrypt. As the key size increases, so does the complexity of exhaustive search to the point where it becomes impracticable to crack the encryption directly.

The most popular encryption cipher is the Advanced Encryption Standard (AES) which is based on the Rijndael cipher. AES with a key size of 256 bits is widely used as it offers the right balance of speed and security. Everything in this group test defaults to this cipher combination.

Besides zuluCrypt, which only uses AES, all others offer additional ciphers. EncFS and eCryptfs also support the Twofish algorithm, while TrueCrypt additionally offers Serpent, and BestCrypt offers Blowfish, CAST, and GOST 28147-89.

Serpent and Twofish are considered by the US National Institute of Standards and Technology to have a higher security tolerance than AES, but are also slower.

TrueCrypt is the only tool that supports cascade encryption, which is the process of encrypting an already encrypted message, either using the same or a different algorithm. It supports five algorithms for this purpose, including AES-Twofish and Serpent-Twofish-AES.

Another security feature that all these encryption tools offer is the use of key files. A keyfile is a file whose content is combined with a password. Until the correct keyfile is provided, no volume that uses the keyfile can be mounted.

Salt data

TrueCrypt also supports adding salt data to the encrypted container. The salt comprises values generated by TrueCrypt's random number generator, and makes it difficult to pre-compute all the keys for a dictionary attack (if a 512-bit salt is used, there are 2^512 keys for each password).

We'd also like to point out that zuluCrypt is a front-end to the cryptsetup utility, which can setup encrypted LUKS volumes based on the **dm-crypt** kernel module. LUKS is the Linux Unified Key Setup, which is a disk-encryption specification designed specifically for Linux. **dm-crypt**, via the cryptsetup command-line tool, can use any algorithm that are built into your kernel which you can find out with the **cat /proc/crypto** command. It can also add salt when creating LUKS containers.

VERDICT

TrueCrypt	★★★★☆
BestCrypt	★★★☆☆
zuluCrypt	★★★☆☆
eCryptfs	★★☆☆☆
EncFS	★★☆☆☆

What can it encrypt?

Disk, partitions, swap…

The encryption tools in this group test can be classified into two types based on how they operate. TrueCrypt, BestCrypt and zuluCrypt perform block device encryption – that is, they encrypt everything written to a certain block device. The block device can be a whole disk, a partition or even a file mounted as a loopback device.

With block device encryption, the user creates the file system on the block device, and the encryption layer transparently encrypts the data before writing it to the actual lower block device. One advantage of this is that attackers learn nothing about the filesystem unless they have the means to decrypt the data. They wouldn't even know the type of filesystem or the directory structure. – while encrypted it just appears like a large blob of random data.

On the other hand, a disadvantage of block device encryption is that a fixed region of storage must be pre-allocated. That's where stacked filesystem encryption solutions, like eCryptfs and EncFS, come into the picture. They add an additional layer to an existing filesystem. Files that are written to an encryption-enabled folder are encrypted on the fly before the underlying filesystem writes them to disk. While they don't need a fixed region of storage, on the downside they don't hide file metadata such as the number of files, directory structure, file sizes, permissions, etc.

Using TrueCrypt and zuluCrypt you can create an encrypted disk within a file or within a non-system partition or USB disk. They cannot encrypt the boot partition or the boot drive in Linux (TrueCrypt supports this feature in its Windows version), but you can use zuluCrypt to encrypt/decrypt any file with either a passphrase or a keyfile.

BestCrypt offers different encryption products but only the Containers app is available for Linux, which creates encrypted containers in which to store files.

VERDICT

TrueCrypt	★★★☆☆
BestCrypt	★★☆☆☆
zuluCrypt	★★★★☆
eCryptfs	★★☆☆☆
EncFS	★★☆☆☆

Noteworthy features

What makes our chosen few stand out from the crowd?

TrueCrypt is the most portable solution of all the tools in this group test, because you can use the application without having to install it. One of its most notable features is the ability to create a hidden volume inside another encrypted volume. Although many experts have played down this feature's effectiveness in concealing the fact that there's a hidden container from a determined attacker, it does provide plausible deniability to casual inspection. Under Windows you can even run a hidden operating system from this partition, though this functionality, as with many of TrueCrypt's specialist features, is unavailable on Linux.

BestCrypt also enables you to create a hidden container inside an already encrypted volume. In fact, with BestCrypt, unlike TrueCrypt, you can create multiple hidden containers. BestCrypt also offers the ability to protect existing hidden containers when creating a new one. Using the tool you can also encrypt the container headers, which conceals the fact that you have an encrypted container. The tool also enables you to re-encrypt containers, change their ciphers and encryption keys, and also lets you add multiple passwords.

Zulus, Sir – faahsands of 'em

zuluCrypt's main attraction is that it lets you manage different types of encrypted volumes, including those created by dm-crypt/LUKS and TrueCrypt. It can also encrypt individual files with GPG.

The one feature common to both stacked filesystem encryption tools is the portability of the encrypted files that can be decrypted on other operating systems as well. Furthermore, eCryptfs lets you assign independently revocable multiple keys for the same encrypted data.

VERDICT

TrueCrypt	★★★★☆
BestCrypt	★★★★★
zuluCrypt	★★★★★
eCryptfs	★★★★☆
EncFS	★★★★☆

Usability

Do you need a superior intellect to use these tools?

Tools that encrypt your data should be handled with care. The FAQs of encryption software deal with accidental data loss because of forgotten passwords, misplaced key files or damaged headers. The same software that guards your data might also prevent you from accessing it ever again.

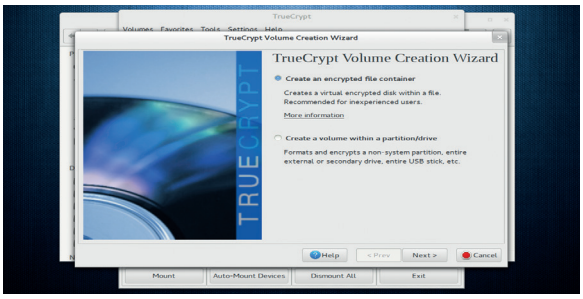
This places even more responsibility on the developers of encryption software to make sure their users aren't overwhelmed by the tools at their disposal. In addition to being easy to install, a well designed tool should expose its features correctly, so that it doesn't force the user to look for the help file or support on the forums.

TrueCrypt ★★★★☆

Because of TrueCrypt's unclear licensing terms, the software isn't available in the repositories of any Linux distro and has to be downloaded from its website in the form a compressed installation script. We also don't like it that TrueCrypt requires you to either use a sudoers configuration or run the tool as the root user.

The app has a graphical interface and can also be controlled from the command line. The step-by-step volume creation wizard makes the tool ideal for new users. TrueCrypt also has an option to benchmark the speed for encryption/decryption of various supported encryption ciphers.

You can use the software to organise volumes and mount them with a single click. The tool also lets you change the encryption password and add or remove keyfiles to volumes.



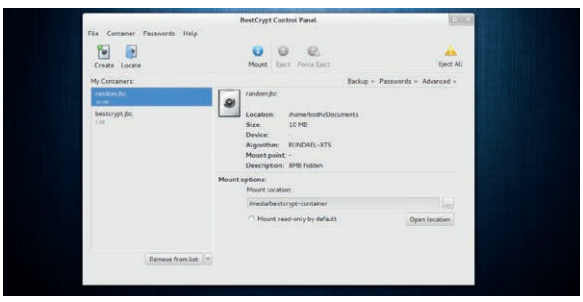
TrueCrypt suffers from its bespoke licence, which keeps it out of distro repositories.

BestCrypt ★★★★☆

The proprietary tool isn't the easiest to install, but its documentation explains the process in detail. Like TrueCrypt, the app has a graphical interface and a command-line utility, which has more functions than the GUI. For example, you can benchmark the performance of the various ciphers supported by BestCrypt only via the CLI.

The graphical interface is pretty simple to navigate. You get buttons to create a new container or load an existing one. New users can create a container simply by specifying a size and selecting the location of the container. Advanced users can optionally bring up additional options to select a filesystem for the container, change its encryption algorithm and encryption mode.

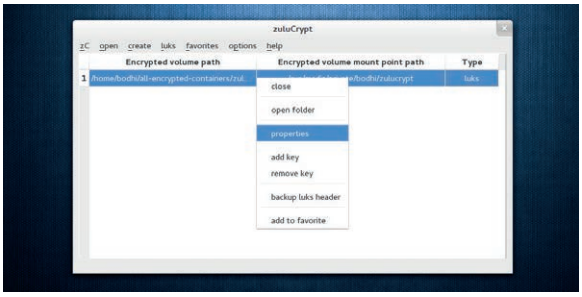
You can perform various tasks on the container before mounting



BestCrypt costs €49.95, and the Linux version doesn't support full-volume encryption.

zuluCrypt ★★★★☆

ZuluCrypt's GUI isn't as polished as BestCrypt's but is more intuitive than TrueCrypt's. You get separate options to create an encrypted container in a file and in a partition. You can also create random keyfiles and use these to encrypt the containers. If you use the app to create a LUKS container, the app will remind you to back up its header immediately after creating the container. The tool also has options to encrypt and decrypt standalone files and securely erase a device by writing random data to it, as well as a graphical tool for mounting and managing volumes.

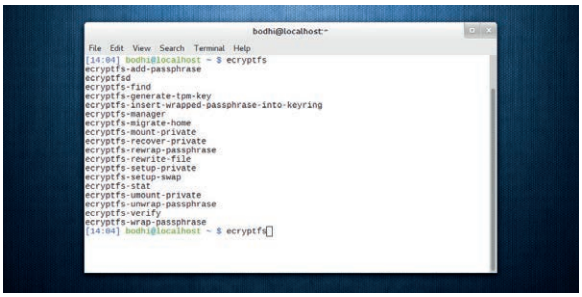


ZuluCrypt is a front-end to the crypsetup utility, but it also has its own CLI version.

eCryptf ★★☆☆☆

The best thing about eCryptfs is that it's available in all major Linux distributions, so you can install the **ecryptfs-utils** package using your distribution's package manager. This will install a bunch of individual utilities to create, mount, and manage all aspects of the file system level encryption system.

When you create a new encrypted directory, the tool will take you through a command-line wizard to help you configure the directory. You'll be asked to select one of the six supported ciphers and its length to encrypt the directory and you can also choose to encrypt the filename within this directory as well.

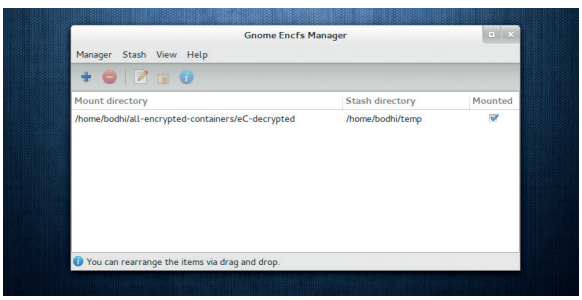


eCrypt is simple to set up: just create a directory that you want to encrypt and mount it as the 'ecryptfs' type. You can also use the utilities to set up a private directory and encrypt the swap.

EncFS ★★☆☆☆

EncFS doesn't have an official graphical front-end, but Gnome users can use the Gnome Encfs Manager to manage and mount directories with it. The EncFS utility comes in several distros, including Fedora.

Setting up an EncFS-encrypted folder is similar to eCryptfs, although instead of using **mount** you need to use the encfs binary. EncFS also requires that two directories are used to keep encrypted and decrypted files. It's common practice to store the encrypted files inside a hidden directory. When you mount a new encrypted directory, you get two pre-configured settings with different encryption settings.



You can invoke EncFS in expert mode, which lets you manually pick the various encryption settings.

Support and documentation

When you're stuck and you need help, where do you turn?

TrueCrypt includes a comprehensive 150-page user guide, which provides detailed instructions on using the application as well as educating users about the precautions they must take when dealing with any kind of encrypted data. There's also a step-by-step beginner's guide and an FAQ that discusses common issues such as forgetting encryption passwords and how to use the software on a removable device. The forum on the website (**forums.truecrypt.org**) has a dedicated board on problems related to the Linux version of TrueCrypt.

When we checked, the latest update was a week ago, whereas the Mac forum had been updated that day. Either Linux users are better than Mac users at using encryption software, or there just isn't as much interest in the Linux version. Or maybe it's a bit of both.

Read the manual, and the FAQ

BestCrypt also has a detailed user guide as well as a quick start guide and a forum board, but no paid support, while the primary source of usage information about eCryptfs is in the form of man pages for the various utilities. It's the same with EncFS, although its website hosts some information about the tool for new users.

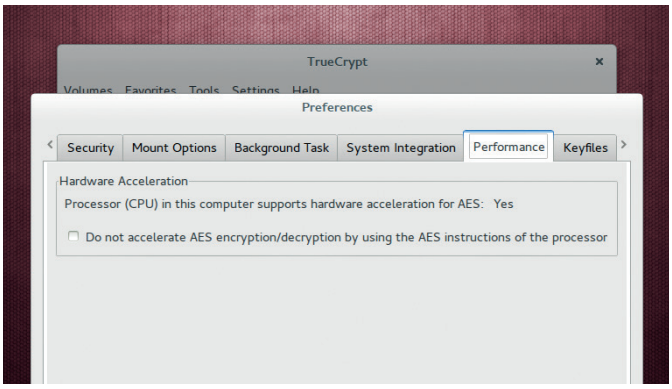
Documentation is the weakest point of zuluCrypt. It has no user guide and no how-tos that explain basic usage of the application. However its developer makes up for that by being quite vocal and active on the popular forum boards. From the Help menu in the app you can find out the default ciphers for the various types of containers you can create with the app and some brief but important information such as the importance of backing up LUKS headers. The FAQ on the zuluCrypt website answers some pretty useful questions and is a must-read for all users.

VERDICT

TrueCrypt	★★★★★
BestCrypt	★★★☆☆
zuluCrypt	★★★☆☆
eCryptfs	★★☆☆☆
EncFS	★★☆☆☆

Performance

Do they add much overhead?



AES is the fastest cipher, with Blowfish just behind. Twofish is relatively slower, and Serpent is the slowest.

TrueCrypt supports parallelised encryption for multicore systems. This means that it can use all the cores in a multicore processor in parallel to encrypt and decrypt the data. Furthermore, header key derivation is also parallelised, which means TrueCrypt can also mount volumes faster. However, one performance enhancing feature in TrueCrypt, pipelined read/write operations, is only available on the Windows version of the tool.

The Quad-Core AMD A8 processor on one of our test machines supports hardware-accelerated encryption, and thanks to this instruction set, which makes the encryption/decryption several times faster than when performed on a purely software implementation. TrueCrypt isn't the only app to take advantage of hardware acceleration. So too can eCryptfs and BestCrypt. However, we couldn't find the option to control the state of the AES hardware acceleration, either in the graphical front-end or the CLI version. There's no information on whether zuluCrypt uses hardware acceleration but cryptsetup does support it.

FUSE boost

Since EncFS ties in to the Filesystem in User-Space kernel (FUSE) module, you should expect some drop in performance when using it. For the same reasons the authors of eCryptFS claim their tool is faster than EncFS because there is no overhead caused by context switching between the kernel and userspace. Many tools have a benchmarking tool built into them to help you compare the performance of the various ciphers for your setup. Some tools, such as BestCrypt, measure performance by averaging the time it takes to encrypt small amounts of data several times, while others let you specify the size of the buffer you wish to encrypt.

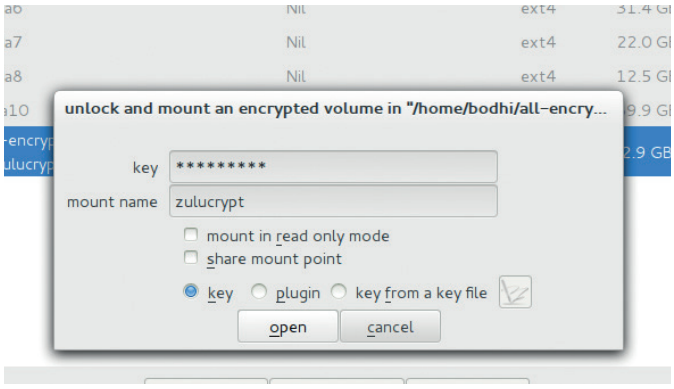
In our tests, TrueCrypt was the fastest, writing over a gigabyte of files in under a minute. eCryptfs was marginally slower while BestCrypt took over three minutes and was the slowest of the lot. EncFS, despite its userspace disadvantage, repeatedly edged out zuluCrypt sometimes by as much as 20 seconds.

VERDICT

TrueCrypt	★★★★★
BestCrypt	★☆☆☆☆
zuluCrypt	★★★☆☆
eCryptfs	★★★★★
EncFS	★★★★☆

Mounting volumes

Working with the encrypted containers.



eCryptfs bundles a script that can use cryptsetup to encrypt the swap partition, as well as the more usual dasa partitions, such as /home.

To mount an encrypted volume you provide the correct password and/or keyfile. Once mounted, an encrypted volume behaves like any other disk. You can even play or record multimedia content, like a video from a mounted encrypted volume – the app will load bits of the video and decrypt it in RAM.

The biggest advantage with both eCryptfs and EncFS is that they can be used to protect existing filesystems without block device access, such as Samba shares or cloud storage folders. They also allow offline file-based backups of encrypted files. eCryptfs has its own set of scripts to mount and unmount encrypted directories. eCryptfs also has utilities that can mount the encrypted directories from an Ubuntu live CD to help you recover data.

EncFS also has its own CLI tool to mount encrypted folders. Like eCryptfs it also needs two directories -- one to hold encrypted data and the other to hold unencrypted data.

You can mount BestCrypt and TrueCrypt encrypted volumes from the graphical interface as well as the CLI. The graphical interfaces of both tools enable the user to mount the volumes as read-only. BestCrypt additionally lets you specify a mount point for the container.

More Zulu excellence

But both are topped by zuluCrypt, which includes the zuluMount tool. This is a general-purpose mounting tool that can mount all encrypted volumes supported by zuluCrypt, including LUKS and TrueCrypt volumes. You can also mount volumes from the main zuluCrypt app, but zuluMount has a simpler interface and is designed with the sole purpose of mounting and unmounting filesystems. In fact zuluMount can mount and unmount unencrypted volumes as well and can even manage plugged-in devices. Like zuluCrypt, the zuluMount tool has a CLI interface as well. zuluMount also lets you make a mount point public and share it with other users.

VERDICT

TrueCrypt	★★★☆☆
BestCrypt	★★☆☆☆
zuluCrypt	★★★★★
eCryptfs	★★★★☆
EncFS	★★★☆☆

OUR VERDICT

ENCRYPTION TOOLS

Block device vs filesystem encryption.

Broadly speaking we have covered two types of encryption tools in this group test. Three do block device encryption and two do stacked filesystem encryption.

You'd use the latter if you wanted to keep an encrypted folder within, say, your home directory. For this purpose you can use either eCryptfs or EncFS. If you want speed, go with eCryptfs, which operates in the kernel space.

The other three tools create encrypted containers. In their encrypted form these file containers appear like unintelligible files, and you can work with them just like any other file on the system, within your file manager.

"zuluCrypt gives you most of TrueCrypt's functionality with a clearer licence."

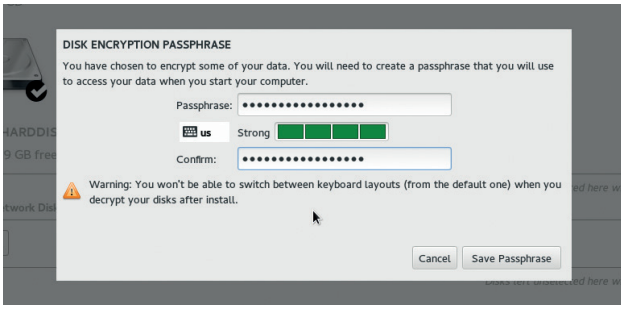
A worthy runner-up

The most popular tool for creating such containers is TrueCrypt. The biggest advantage of the app is its portable nature, which means you can run in without installing the app. This is truly a useful feature especially when combined with the fact that the app is available on multiple-platforms. In effect you can bundle TrueCrypt executables in a removable device along with the encrypted containers and decrypt them under any OS.

The app also offers denial encryption just like BestCrypt. But the use of hidden folders can be dangerous in the hands of inexperienced users: any changes you do to the mounted main part can overwrite and damage the hidden part.

All things considered, zuluCrypt comes out on top, as it has several distinct advantages over TrueCrypt. To begin with, the app doesn't expect you to have a sudo setup like TrueCrypt. Also with zuluCrypt you get most of TrueCrypt's functionality with a clearer licence. zuluCrypt can create encrypted volumes in both files and partitions and allows the use of keyfiles.

zuluCrypt can also encrypt individual files and can read different types of containers. To top it off, it has a nice intuitive graphical interface and a specialised tool for managing encrypted and unencrypted partitions.



If you are serious about encryption, pick a distro that offers the option to encrypt all contents in your disk.

1st zuluCrypt

Licence GNU GPL v2 **Version** 4.6.7

http://code.google.com/p/zulucrypt
Built on a solid foundation, the tool's intuitive graphical interface makes up for the lack of documentation.

2nd TrueCrypt

Licence TrueCrypt Licence **Version:** 7.1a

www.truecrypt.org
One of the most popular encryption tools currently undergoing an audit to iron out some long pending issues – such as its licensing.

3rd eCryptfs

Licence GNU GPL v2 **Version** 103

http://ecryptfs.org
One of the strengths of this tool are its wonderful utilities that can also help locate and recover encrypted data.

4th EncFS

Licence GNU GPL v2 **Version** 1.7.4

www.arg0.net/encfs
The lack of an official graphical tool is made up by the availability of a pretty good third-party one.

5th BestCrypt

Licence Proprietary **Version** 2.0-3

www.jetico.com
The proprietary tool doesn't offer anything worth recommending over its open source competitors. Did we mention it costs €49.95?

YOU MAY ALSO WISH TO TRY...

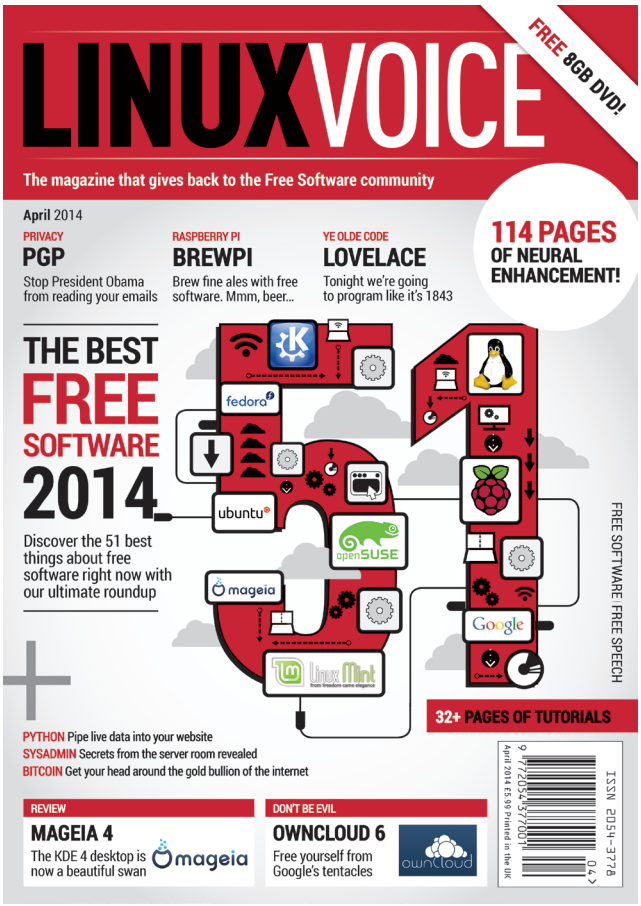
zuluCrypt is a front-end to the cryptsetup and tcplay command-line utilities. As we've mentioned earlier in the group test, the cryptsetup utility lets you create encrypted volumes based on the dm-crypt kernel module. Then there's tcplay, which is a feature-rich BSD-licensed implementation of TrueCrypt. If you prefer you can use these utilities without the zuluCrypt GUI – that is, direct from the command line.

dm-crypt/LUKS can be applied to any type of device that is natively understood by the kernel. It can encrypt whole disks, removable media, partitions, software RAID volumes, and logical volumes. It can also encrypt the swap partition and in fact the eCryptfs tool uses the utility to encrypt the swap partition. It can also encrypt files as long as they mounted as a loopback device (with the losetup utility), and thus available under the /dev directory. cryptsetup can take advantage of accelerated encrypted hardware, and you can format the container with any filesystem that's supported by the kernel.

SUBSCRIBE

shop.linuxvoice.com

Not all Linux magazines are the same



Introducing Linux Voice, the magazine that:

- Gives 50% of its profits back to Free Software
- Licenses its content CC-BY-SA within 9 months

12-month subs prices

UK – £55
Europe – £85
US/Canada – £95
ROW – £99

6-month subs prices

UK – £38
Europe – £53
US/Canada – £55
ROW – £60

DIGITAL SUBSCRIPTION ONLY £38

Each month Linux Voice includes 114 pages of in-depth tutorials, features, interviews and reviews, all written by the most experienced journalists in the business.

Payment is in Pounds Sterling. 12-month subscribers will receive 12 issues of Linux Voice a year. 6-month subscribers will receive 6 issue of Linux Voice. If you are dissatisfied in any way you can write to us to cancel your subscription at subscriptions@linuxvoice.com and we will refund you for all unmailed issues.

NEXT MONTH IN

LINUX VOICE

ON SALE THURSDAY 27 MARCH



CODE FOR ALL

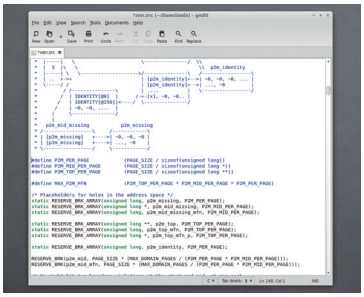
Bring your evil genius to life on every kind of Linux: Firefox OS, Android, desktop Linux – even QML, so you can have your apps ready when the Ubuntu phone roars into life!

EVEN MORE AWESOME!



Old Code

We stand on the shoulder of giants: none broader than Rear Admiral Grace Hopper, the brain behind UNIVAC and filer of the best bug report ever.



Hack the kernel

The impenetrable mysteries of the Linux kernel are ours to command. Well, we can perform the kernel equivalent of 'Hello, World', which is still pretty cool.



Munich migration

Find out how Munich city council switched 15,000 Windows desktops to its own custom Linux and made Steve Ballmer even angrier than he normally is.

LINUX VOICE IS BROUGHT TO YOU BY

Editor Graham Morrison
graham@linuxvoice.com
Deputy editor Andrew Gregory
andrew@linuxvoice.com
Technical editor Ben Everard
ben@linuxvoice.com
Editor at large Mike Saunders
mike@linuxvoice.com
Creative director Stacey Black
stacey@linuxvoice.com

Editorial consultant Nick Veitch
nick@linuxvoice.com

All code printed in this magazine is licensed under the GNU GPLv3

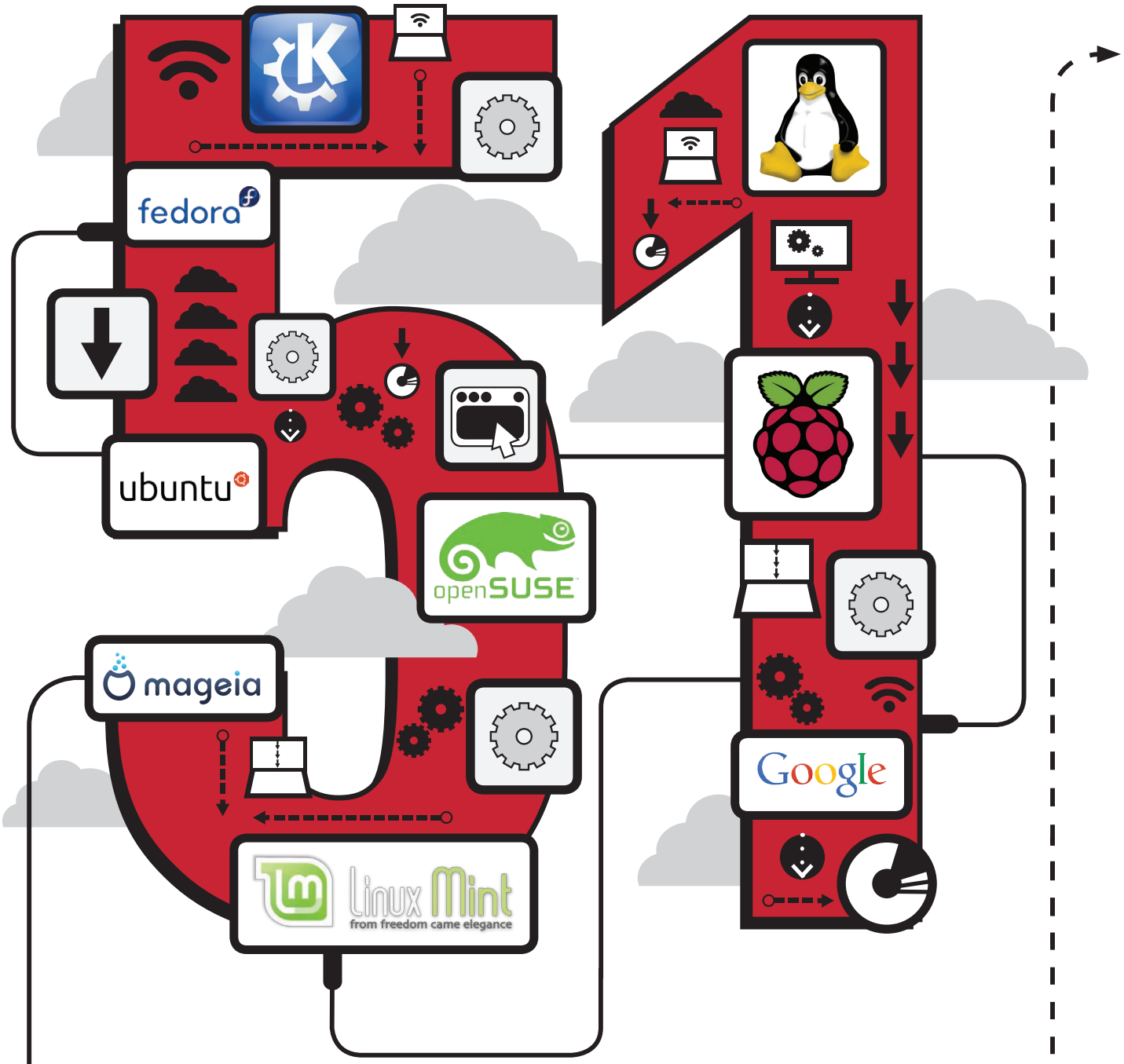
Printed in the UK by
Acorn Web Offset Ltd

Disclaimer We accept no liabiliy for any loss of data or damage to your hardware through the use of advice in this magazine. Experiment with Linux at your own risk!
Distributed by Marketforce (UK) Ltd, Blue Fin Building, 110 Southwark Street, London, SE1 0SU
Tel: +44 (0) 20 3148 3300

Circulation Marketing by Intermedia Brand Marketing Ltd, registered office North Quay House, Sutton Harbour, Plymouth PL4 0RA
Tel: 01737 852166

Copyright Linux is a trademark of Linus Torvalds, and is used with permission. Anything in this magazine may not be reproduced without permission of the editor, until November 2014 when all content (including images) are re-licensed CC-BY-SA.
©Linux Voice Ltd 2014
ISSN 2054-3778

Subscribe: **shop.linuxvoice.com**
subscriptions@linuxvoice.com



OF THE BEST THINGS ABOUT FREE SOFTWARE 2014

This year will be the best ever for Linux and Free Software. Why? **Mike Saunders** has 51 reasons...

For so long, it looked like Linux was on the cusp of causing a revolution, of pulling the masses away from Windows and introducing them to a new world of computing. Linux distributions were becoming easier to use by the month, desktop applications were being refined and polished, and PC vendors were starting to ship Linux with their machines.

While the almighty annihilation of Windows never happened, a much more subtle set of changes took place. Linux is everywhere now – but not many people know it. GNU/Linux powers many of the biggest websites in the world, serving up web applications to hundreds of millions of users. Android dominates the mobile scene, with countless people carrying little Linux machines in their pockets. And the Raspberry Pi has been a storming success, introducing children worldwide to open computing.

2014 has a huge amount in store for Linux users, and not just in terms of software – people, communities and events will also shape the course of the year. So read on for 51 awesome things to look forward to...

1 FEDORA 21

Fedora 21 bucks the trend of previous releases by having a longer development cycle than usual which will hopefully provide more time for new technologies like Wayland to settle down. All being well, the distro will arrive in August. And the lack of a codename is significant: some would argue that names like "Beefy Miracle" stopped certain users from taking Fedora seriously.



Recent Fedora releases have been (in) famous for their quirky codenames; it looks like Fedora 21 will be a lot more serious.

2 FIREFOX VERIFIED BUILDS

Even if you're running an open source browser, you can't be 100% certain that the binary executable doesn't have an NSA backdoor inserted, possibly via a rogue compiler. Well, this is changing: Mozilla is pushing ahead with Firefox verified builds, which means you'll be able to prove that nothing dodgy has been added to the source code before it was compiled.

3 FREEBSD 10

FreeBSD is more conservative than Linux in that it doesn't get fancy new features so quickly, but it's more stable in the long run. FreeBSD 10 has just been released, with the GCC compiler suite being replaced by Clang in the base system, Raspberry Pi support, and a new bhyve hypervisor. See **https://wiki.freebsd.org/WhatsNew/FreeBSD10**.

4 GIMP 2.10

Talk about development hell: GEGL, the Generic Graphics Library for Gimp, has been in development since 2000, yet still isn't an official part of the program. GEGL will support images with higher bit depths, along with non-destructive editing. Parts of it were implemented in Gimp 2.6, and we should see the whole shebang rolled into Gimp 2.10.



Brno has Akademy, good beer and horses – what's not to like?

5 KDBUS

Lennart Poettering is at it again. After causing a stir among boot script and sound server developers with systemd and PulseAudio, the German coder is now trying his hand at inter-process communication (IPC) in the Linux kernel. Right now, applications and background processes can communicate with one another via D-Bus, which Poettering describes as "fantastic". But if it's so good, why do we need to replace it?

For starters, D-Bus is fine for sending short messages between programs (eg a desktop volume applet telling the sound server to knock it down a notch), but it's not so good for transporting data, as it adds latency and complication. Kdbus (Poettering's project) is an in-kernel version of D-Bus, designed for shifting large (multi-gigabyte) quantities of data with minimum overheads. It's still very much in development, but providing it gets a (usually very rare) thumbs-up from Linus Torvalds, we might see it in the mainline kernel tree by the end of the year.

Poettering is developing Kdbus with long-time kernel hacker Greg Kroah-Hartman and others; follow its progress here: **https://github.com/gregkh/kdbus**.

INKSCAPE 0.91

Hang on a minute – hasn't Inkscape been at 0.48.x for years now? Well, yes, but we've always thought this was unfair. Inkscape is an excellent vector editor already used in production environments, so we're glad to see the next release will get close to the magic 1.0 number. Version 0.91 will include: Cairo rendering for the display engine and PNG export; a new grayscale display mode; and heaps of performance improvements. It'll use 25% less RAM in some cases too, making it better for older hardware.

6 AKADEMY

Free software events are great. There's lots of hacking, lots of brainstorming, and lots of beer. Akademy is the annual summit for KDE developers: it's a free and non-commercial event organised by the community. This year it will take place from 6–12 September in Brno, second city of the Czech Republic, and a beautiful place at that. New features will be proposed and discussed, so head to **http://akademy.kde.org** if you want to take part.

7

DEBIAN 8.0 FREEZE

It looks like we won't see Debian 8.0 this year, which isn't surprising given the distro's long gaps between releases. One of the biggest changes will be the switch to a new boot-up system: the old init scripts will be replaced by systemd or Upstart (as used in Ubuntu). A debate is taking place about the possible boot systems, and Debian 8.0 won't be frozen (closed for new features) until November, so there's lots of time to decide.

EDWARD SNOWDEN

To some he's a true American patriot, fighting to defend the US constitution against attacks by a spy-crazy government. To others he's a thrice-cursed traitor who revealed confidential intelligence information to the rest of the world, harming American interests. In any case, there's no doubt that Edward Snowden's leaks have caused a huge debate – not just in the US but around the world.

Whether it makes us safer or not, the sheer amount of spying on their citizens that governments are doing is shocking. Edward Snowden hasn't come out waving Linux flags and championing open source, but a lot of people are starting to ask: can I really trust my software? Can I be sure that Microsoft, Apple, Google *et al* are really doing their best to defend my data? How much are these companies sharing with governments?

Linux and Free Software is no panacea, but at least the openness of its source code provides reassurance that dodgy back-doors added by intelligence agencies would be spotted quickly. The NSA revelations provide useful ammunition for Linux advocates: we can confidently say "Our software is very unlikely to be spying on us, because we can read the source code".

There are more Snowden leaks due in 2014, and they will drive more people to investigate Linux and Fre Software. There's even a Snowden tribute distro: **www.binaryemotions.com/snowden**.

BANSHEE 3.0

Hopefully we'll see version 3.0 of this awesome music player very soon: it's being ported to GTK 3, and will also support synchronisation with Symbian devices. 37 bug fixes have been made since the 2.6 release too.

ENLIGHTENMENT E19

The next release of the world's fanciest window manager will bring a new compositor and better Wayland integration. Lead developer Rasterman hopes to "close the feature window sometime in February", so we're likely to see the final release hit the internet some time in the summer. **www.enlightenment.org**

DIGIKAM 4.0

Arguably the best open source photo manager in existence, Digikam is going from strength to strength. The 4.0 release – due to be released in May – brings oodles of new features thanks to the Google Summer of Code. Working with tags will be much easier via a new hierarchical tags manager and drag-and-drop capabilities in the Tags Manager, while Pick Labels can be used to auto-tag images. Nepomuk support has been re-implemented, after being broken for a while. **www.digikam.org**

GNU HACKERS' MEETING 2014

The GNU Hacker's Meeting is an event to discuss technical, social and organisational issues relating to Free Software and the GNU Project. It will take place on 15–17 September in Munich at the Technical University, and while it's free to attend, you have to register, at **www.gnu.org/ghm/2014**.

Anyone can give presentations ranging from 30–60 minutes, so if you've been working on a piece of software you'd like to see in GNU, let the community know.



Edward Snowden's leaks are encouraging some people to try Free Software.
image: Laura Poitras / Praxis Films

SOFTWARE FREEDOM DAY

We all try to spread the word about Linux and FOSS, but sometimes it's difficult, especially over the internet. So every September, Free Software supporters organise real-world events to promote the benefits and values of FOSS, and everyone is welcome to get involved. You could set up an installfest, for instance, where curious Windows or Mac users bring along their machines and you set them up with a newbie-friendly Linux distribution.

Alternatively, you could give a presentation explaining why Free Software is good for society and encourages sharing. It's a great way to demonstrate that our community isn't just a gaggle of geeks fiddling with code via the internet – we're real people trying to make the world better.

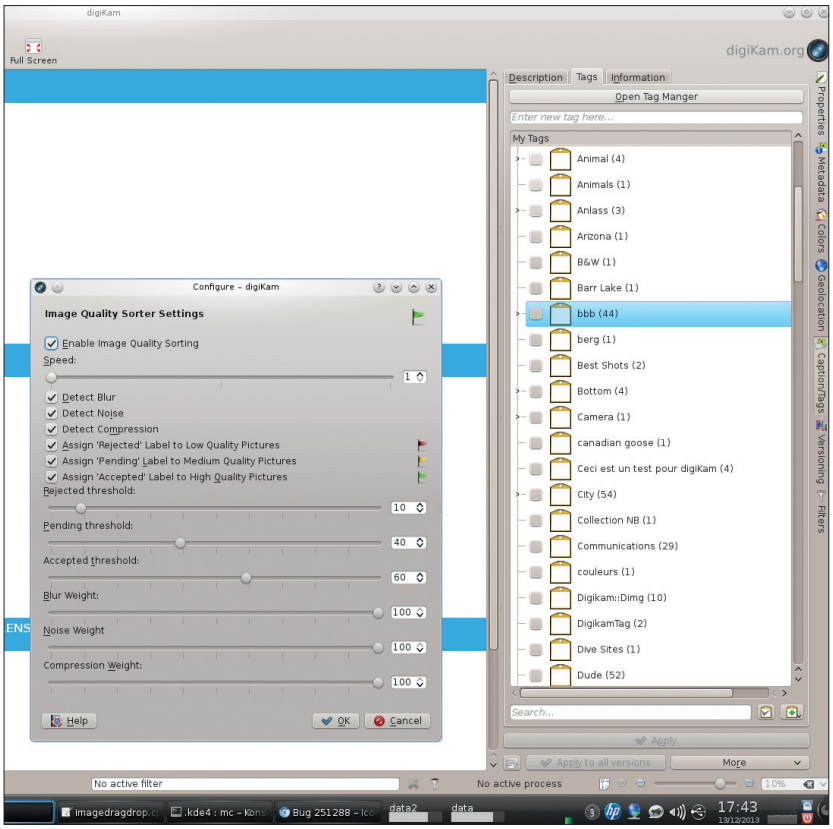
This year's Software Freedom Day will take place on 20 September, so to start your own event or join an existing team, visit **www.softwarefreedomday.org**.



2013's Software Freedom Day saw a huge number of events across the whole globe.

EMBEDDED LINUX CONFERENCE

Cars, washing machines, fridges, doorbells – you name it, Linux is in it. The development scene for embedded Linux devices is bustling, so if you'll be in San Jose from 29 April – 1 May, check this event out. http://tinyurl.com/4gvlozz

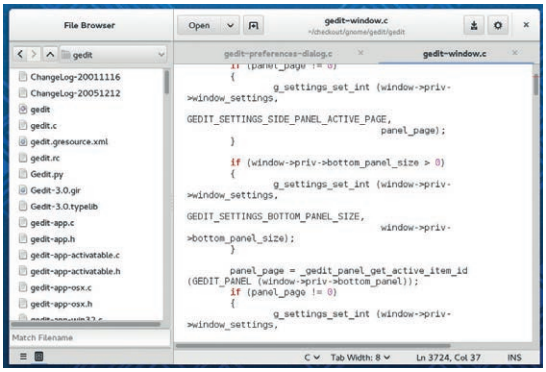


Digikam 4.0 will feature much better tag management, and support for Nepomuk.

MIR

Few things in the recent history of Linux have been as controversial as Mir. Ubuntu's replacement for the X Window System was met with widespread hostility: why didn't Canonical go with Wayland, like everyone else? Was this a symptom of "not invented here" syndrome, or is Canonical trying to distance itself from the Linux mainstream?

Various technical reasons for the existence of Mir were put forward, and Canonical hoped to have Mir running by default in Ubuntu 14.04. This didn't happen, so maybe we'll see it in Ubuntu 14.10, but the controversy continues. An Intel developer working on an open source graphics driver recently removed support for XMir, a Mir compatibility layer for X applications. Intel's reasoning? "We do not condone or support Canonical in the course of action they have chosen". Some Intel developers are working on the Wayland project, so there's plenty of politics involved.



Say hello to Gedit's new interface, debuting in Gnome 3.12. Like Gnome 3, it might take some getting used to…

GNOME 3.12

Scheduled for release at the end of March, and due to land in the major distros shortly after that, Gnome 3.12 will be largely an incremental rather than a major improvement. That's fine by us: plenty of people are still trying to get their heads around the Gnome 3 interface, so the more fine-tuning the developers do, the better. Overall performance will be boosted thanks to Gnome Shell becoming a single binary file rather than a bunch of JavaScript files, while the Epiphany web browser has seen some interface polish.

Gnome Software will support a rating system along with the ability to launch installed apps, and the Gnome Online Accounts tool receives support for online bookmarking service Pocket. If you're a regular user of Gedit, be prepared for a big revamp of the interface. There's already a lot of debate about the UI changes, but you'll be able to try it for yourself very soon.

GCC 4.9

The next version of the GCC will include support for OpenMP 4.0, along with Intel's Silvermont and Broadwell microarchitectures. Our favourite feature? Coloured warnings and error messages!

HAIKU OS R1?

We've been following the progress of Haiku OS, an open source BeOS clone, for as long as we can remember. The goal is noble: a lightning-fast, trimmed-down, multimedia-friendly OS built specifically for the desktop (so it doesn't get sidetracked trying to be a server OS as well). As much as we love Linux on the desktop, it's always good to have some competition – and Haiku is starting to provide it.

It's still in the alpha stages of development, but wrinkles are being ironed out and more native software is arriving all the time. We hope to see the first beta release this year, and maybe the first full one (R1) as well, if we're lucky. **www.haiku-os.org**

STEAMOS

It's going to be a huge year for gaming on Linux. Until very recently, PC gaming was almost entirely the domain of Windows, with just a tiny slither of triple-A titles making their way to our choice of operating system. Valve's decision to use Linux for its Steam Machines has caused quite a stir, though: suddenly Microsoft isn't involved at all. SteamOS isn't a typical distro and won't cause a mass influx of Windows users to Ubuntu or Fedora, but it will show that Linux is a great OS for gaming.

SCRIBUS 1.5

Scribus, the open source desktop publishing app, has been at version 1.4.x for over two years now, so we're eagerly awaiting the 1.5 release. It will bring tabs for working on multiple documents, a new Preferences dialog, support for more colour palette formats, and a new picture browser.
http://wiki.scribus.net/canvas/1.5.x_Roadmap

LINUXTAG 2014

Germany's biggest Linux show (it's been running since 1996) will take place from 8–10 May in Berlin. Droidcon, the "world's largest Android developer event", will also be housed under the same roof, so we can expect a feast of Linux-related fun in Germany's counter-culture capital
www.linuxtag.org/2014/en

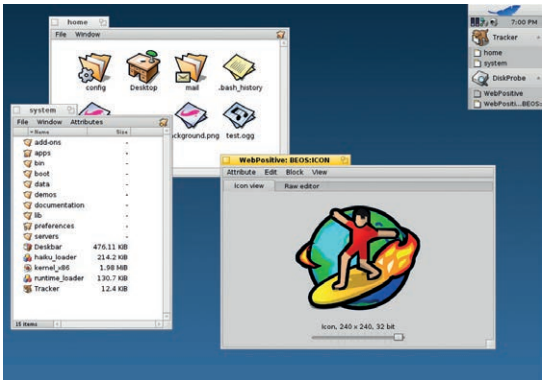
CENTOS AND RED HAT

For years, Red Hat quietly accepted the existence of CentOS, a free rebuild of its Red Hat Enterprise Linux (RHEL) product. Linux admins could do testing work on CentOS builds, and pay Red Hat for commercial RHEL support subscriptions later if necessary. The two projects co-existed pretty well, and now they're going to work together.

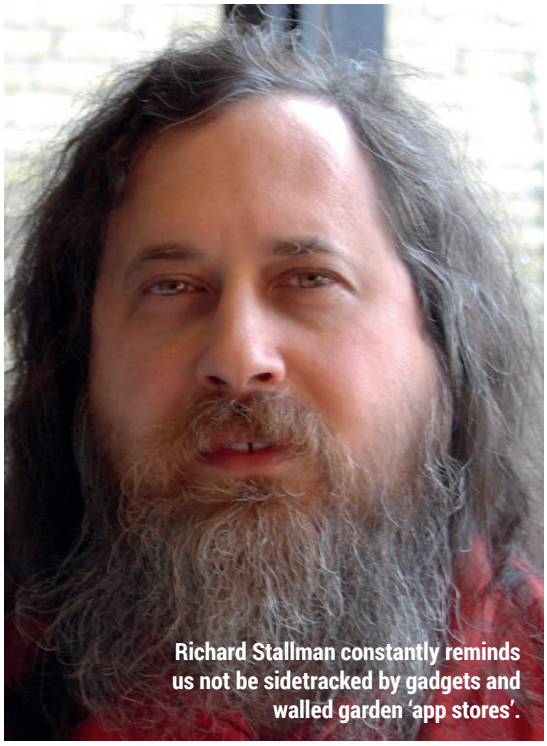
Red Hat is employing a bunch of CentOS developers who will continue to work full-time on CentOS, but who will also enjoy greater integration with the RHEL and Fedora communities. It's a bold move from Red Hat, but it makes sense: CentOS users may not pay money to Red Hat now, but a healthy CentOS community means more potential RHEL customers in the future.

ROOTLESS X

Currently, some parts of the X Window System run as root, which opens up potential security vulnerabilities. Hans de Goede has patched X to work with systemd-login and obviate its need for root privileges entirely. Maybe we'll see it in distros this year...



We're (still!) really excited about Haiku, a streamlined OS built from the ground-up for desktop use.



Richard Stallman constantly reminds us not be sidetracked by gadgets and walled garden 'app stores'.

GOBOLINUX 015

After a five-year hiatus, GoboLinux is back. Chances are you've never heard of this distro, but it's rather cool: it has an entirely different filesystem hierarchy to other distros. Instead of program files being scattered across /usr/bin, /usr/lib, /usr/share and so forth, in GoboLinux every program lives in its own directory. This makes it much easier to manage installations by hand and copy programs to other machines.
www.gobolinux.org

RICHARD STALLMAN

Love him or loathe him, Richard Stallman has a habit of getting things right. The founder of the GNU project (and Free Software Foundation) has been known to launch into rants about topics that don't seem immediately relevant, but a few years down the line, we all end up scratching our heads and thinking, "Hmm, RMS was right". His uncompromising stance on freedom makes him hard to deal with at times – but he has a knack of spotting problems way down the line.

Way back in 1997 he wrote a parable called The Right to Read (**www.gnu.org/philosophy/right-to-read.html**), which described a futuristic world where the sharing of books and written information is punishable by law. Here we are, 17 years later, and DRM-laden eBooks are being sold by the millions. Customers are buying materials that they can't share, sell or pass on to their kids without potentially landing themselves in legal trouble.

Stallman saw this way down the line, so it will be interesting to see what topics he brings up this year. Even if he talks about threats to our freedom that don't seem looming right now, you can bet your bottom dollar/pound/euro that they'll come up in the future.

MAGEIA 4

We still can't agree on how to pronounce it, but this Mandriva spin-off is doing well, and version 4 will deliver a new welcome screen, software updates, and GTK 3 ports for most of its drak*/*drake configuration tools.
www.mageia.org

PARALLELLA

This dinky credit card-sized computer has a lot in common with the Raspberry Pi, but it's designed for developing high-performance, parallel processing applications. It comes with a 16- or 64-core Epiphany co-processor, and starts at $99.
www.parallella.org

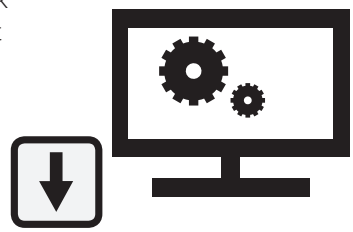
OPENSUSE 13.2

We should have a new release of OpenSUSE to play with in July. It's still early in the development cycle, but it's planned that the installer will be simplified, and Plasma-nm should replace NetworkManager-kde4. The new installer could make its way into SUSE Linux Enterprise Server 12 as well, which should arrive around the middle of the year.

XFCE 4.12

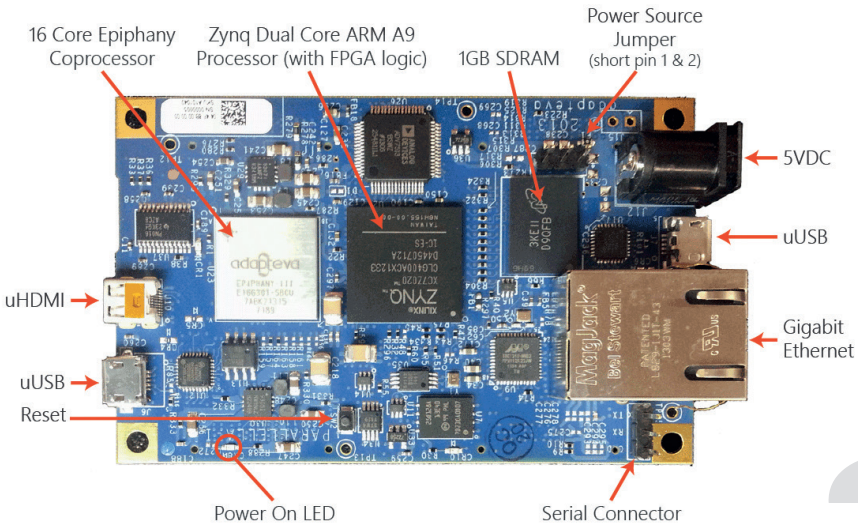
Xfce picked up a bunch of new users in the last couple of years, many of whom left Gnome after the radical Gnome 3 redesign. Development is rather conservative in the Xfce camp: the last major release arrived in April 2012, and Xfce 4.12 won't actually move to GTK 3, as some people expected. Instead it will stay with GTK 2, but put the framework in place to move to the newer toolkit at a later date.

Feature-wise, Xfce 4.12's window manager (xfwm4) will sport a new smart placement mode, which has been rewritten to be more clever than the previous version and align windows next to each other (instead of putting them in a random gap). Thunar, the file manager, will support showing properties for multiple files simultaneously, and also show mounted remote locations in the shortcuts pane.



HURD 0.6?

Development on GNU's own kernel has been stagnant for many years, but the recent release of 0.5 could spur a bit more hacking activity. HURD is a microkernel that aims to be more secure and stable than microkernels like Linux. Maybe 0.6 will arrive this year: www.gnu.org/software/hurd/hurd.html



Parallella: it's like the Raspberry Pi, but with a 16/64-core co-processor for parallel processing applications to bump up the power (and the price tag).

KERNEL 3.14...

Kernel 3.13 has only just been released as we write this, but plans are already underway for 3.14. The zRAM filesystem, which creates compressed RAM filesystems (eg for use as high-performance swap areas) is now in the official kernel source tree, and we'll see improved Intel Broadwell microarchitecture support. Random number generation should be even more random and performance of the SquashFS filesystem has been boosted.

Other candidates for inclusion in 3.14 are enhancements to power management on Intel chips, support for dynamic refresh-rate switching in Intel's DRM driver, and Nvidia Tegra Prime support.

...KERNEL 4.0?

While announcing kernel 3.12, kernel maintainer Linus Torvalds said that he expects 4.0 to follow 3.19, probably some time this year. It won't be a big change – just to avoid the "crazy [version] numbers we had in the 2.x series". Stability will be a priority.

LIBREPLANET 2014

On 22 and 23 March in Cambridge, MA, USA, the LibrePlanet conference will assemble activists and developers to discuss challenges to software freedom. This year's topic government and corporate surveillance. **https://libreplanet.org/2014**

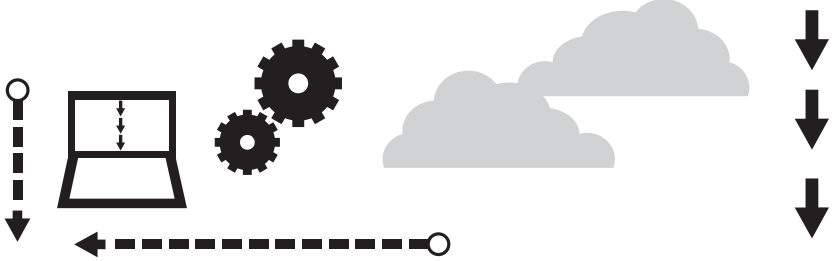


LibrePlanet 2013 was a great success, with 150 attendees and 35 talks, workshops and events. (Photo CC-BY-SA, Free Software Foundation).

KDE

There are so many great KDE applications out there, but if you use a different desktop, they often don't fit in especially well with your graphical environment. Typically they expect various KDE services to be running and depend on lots of KDE-specific libraries, bumping up RAM consumption. KDE Frameworks 5 aims to fix this by moving lots of KDE's functionality into the Qt libraries – so pure Qt applications can benefit from this too.

This means that many current KDE apps can become Qt apps, making them simpler to install and port to other platforms. A beta release is planned for April, with the final release scheduled for June, but as always these dates can slip. The last major KDE architecture change (from 3.x to 4.x) wasn't well received at first, so hopefully this time the process will go more smoothly, and it won't take a couple more years before everyone is happy again.



X.ORG SERVER

Due in July, the next major update for the X Window System could include support for GLAMOR, an OpenGL-based acceleration library that speeds up 2D operations. In addition, veteran X developer Keith Packard has reduced the build warnings from 1,047 to zero. Nice one.

MAILPILE

Sick of Google, Microsoft and governments reading your emails? Looking for an alternative? Mailpile is a self-hosted, ad-free and open source email system with encryption built in. It's still undergoing heavy development, but an alpha release is due very soon. **www.mailpile.is**

MACBOOK PRO SUPPORT

Apple is the antithesis of Free Software in many respects: the company is secretive, makes proprietary software, and pushes developers into its highly controlled app store. Go to any major Linux conference, though, and you'll see lots of MacBooks running Linux: the hardware itself is well regarded for performance, weight and battery life. Kernel 3.13 brings lots of fixes for Retina MacBook Pros, fixing sound, display and power management issues. If you like Apple hardware but want to avoid OS X, grab a distro with the new kernel.

BTRFS

Btrfs (the B-Tree filesystem) is going to be big: it's jointly developed by Red Hat, SUSE, Intel, Oracle and other well-known companies. It's still marked as experimental, but the cool features it boasts (transparent compression, online resizing and snapshots, partitions spanning multiple drives) mean that it's getting a lot of attention. Some of the major Linux distros could move to Btrfs as the default filesystem this year, providing it's stable enough.

GUADEC 2014

This year's Gnome Users and Developers Conference takes place in Strasbourg, France, from 26 July until 1 August. The schedule hasn't been decided just yet, but there will be talks, presentations, hackfests and parties. **https://wiki.gnome.org/GUADEC/2014**

CRUNCHBANG 12

It's fast, it's light, it looks slick and it has Debian underpinnings: CrunchBang is an awesome distro. Version 12 will be based on Debian 8 (Jessie) sources, and lead developer Corenominal is looking for new ideas on the forums: **http://tinyurl.com/qdbkvta**



It's still early days for Firefox OS, but hopefully it will pump some innovation into the mobile OS market.

FIREFOX OS

We're very happy to see more competition in the mobile operating system space. Android is great, but if the market ends up dominated by just Android and iOS, things could get very stagnant. Firefox OS is currently targeted at low-end devices, and enables developers to write 'native' web apps by using APIs that communicate with the phone's hardware.

Right now there are very few devices running Firefox OS, but the range is slowly growing. Version 1.4, due in May, should bring: creation of ringtones from songs in the music app; sharing ringtones via Bluetooth and other protocols; support for NFC payments; application switching via edge gestures; and remote wiping for security.

Keep an eye on **https://wiki.mozilla.org/B2G/Roadmap** to see how the 1.4 release unfolds, and by the time you read this, the first Firefox tablet (the snappily named InFocus New Tab F1) may be available for aspiring developers to have a play with.

MINETEST

Minecraft is one of the most addictive games ever made – and in fact, it's unfair to call it a game. It's a giant construction engine, a world simulator, and a way of life. Minetest, an open source clone, is coming along well although it's still lacking many features and only at version 0.4.9 right now. Future releases this year should see a new map generation system, with more variation between biomes (landscape types). **www.minetest.net**



Explore and rebuild a gigantic virtual world with Minetest.

44 REACTOS

Creating a fully Windows-compatible open source operating system is a mammoth task, but the ReactOS team is chipping away at it. Version 0.4 will include better networking, sound and USB support, and is due to arrive this year. **www.reactos.org**

45 LIBREOFFICE 4.2

The current major release of LibreOffice arrived at the start of February, so it should be in your distro by the time you read this. We're always impressed by the effort that goes into LibreOffice releases, and 4.2 is no exception: it's a goody bag of new features.

You can now format individual characters with borders in Writer, while the spelling-checker pop-up menu now lets you do change tracking operations. Additionally, you can export **.dot** (MS Word document template) files. Calc now has a random number generator, together with statistics functions for data analysis. In Impress, a new icon is shown in the Slide Sorter if a slide has a transition or animation effect, while integration with Gnome 3, MATE and Xfce has been improved. Then there's an Expert Config panel in the Options dialog for under-the-hood tweaking.

46 LLVM/CLANG

GCC has been the *de-facto* standard free compiler suite for years, but LLVM/Clang is catching up, offering a more modular design. Work is underway to make it compile the Linux kernel, and perhaps we'll even see a Clang-compiled distribution before the year is out…

47 SAILFISH OS

It's still a baby and only available for one phone, but Sailfish could be a promising mobile OS in 2014. Built on a Linux kernel with Wayland and Qt providing the interface, Sailfish looks pretty, but it will have to work hard to avoid becoming another Maemo/Meego/Moblin/Tizen-like abandonware.



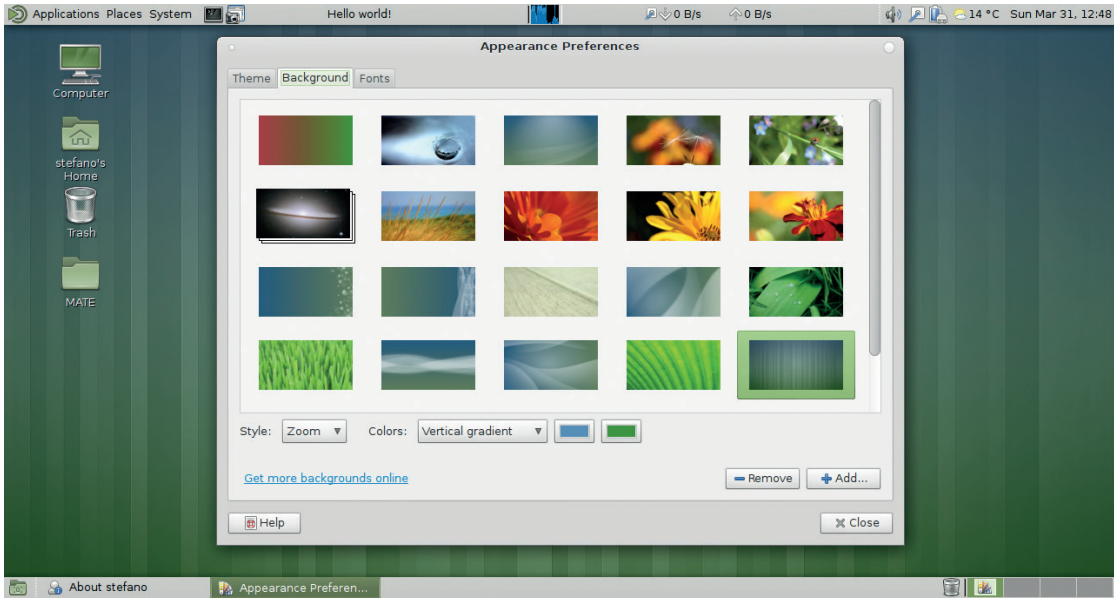
Elementary OS is still undergoing heavy development, but it already looks deeply gorgeous.

48 MATE 1.8

This fork of Gnome 2 has done better than many expected, with a healthy community building up around it. Version 1.8 will add support for G-Streamer 1.0, panel background rotation, and the ePub format in Atril (a fork of the Evince document viewer). In addition, work is underway to add support for Wayland and the AccountsService from freedesktop.org. MATE 1.8 was originally planned to be included in Mint 16, but we'll have to wait a bit longer for it now – it'll be worth it.

49 ELEMENTARY OS

One of the prettiest distros we've ever seen, Elementary OS's website (**www.elementaryos.org**) is just as polished as the software. This isn't just yet-another-distro with some glitz sprinked on top; it sports its own applications. Another beta is due soon.



Some assumed that the MATE desktop wouldn't last – but it's thriving, with version 1.8 just around the corner.

50 LINUX PLUMBER'S CONFERENCE

Everyone loves talking about wobbly windows and desktop apps, but this conference is for hackers who work on the guts of Linux: kernel subsystems, core libraries and so forth. It'll be held from 15–17 October in Düsseldorf, Germany. **linuxplumbersconf.org**

51 YOU

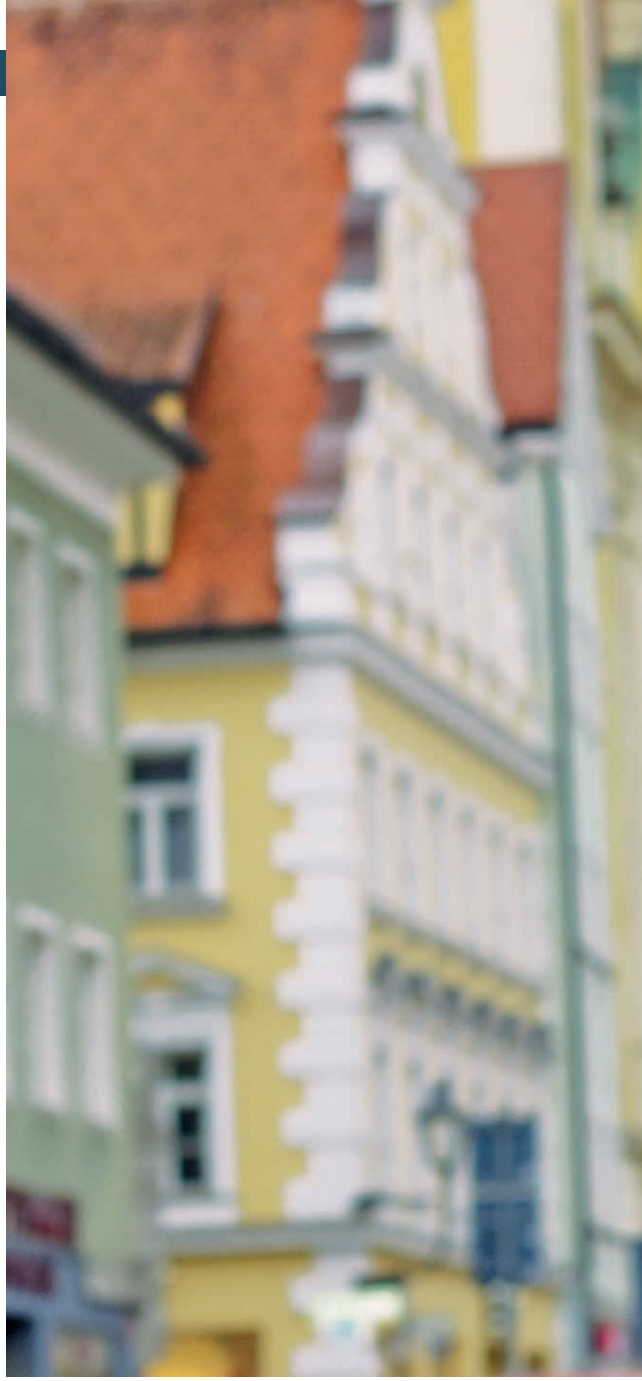
Sure, it may be a bit cheesy to end with this, but it's true. It's you, the Linux community, that will make this an awesome year. Every little thing you do, whether it's helping a newbie on a website forum, or submitting a bug report for an app, adds up to make the Linux and Free Software ecosystem even better. There are times when it's frustrating, when progress seems to be slow and arguments break out, but providing we stick together, explain the advantages of Free Software and spread the word positively, we will succeed in the long run.

So, a hearty thank you to everyone in the Linux community – every user, developer, documenter, designer and tester. Great times are to come in 2014!

THE OFFICE CRUSADER

We approve of projects forking to do a better job – and one of the best examples is LibreOffice…

For our first magazine interview, we got some cheap flights and headed out to Kaufbeuren, an attractive Swabian city an hour's train ride from Munich. This is where we met Florian Effenberger, chairman of the board at The Document Foundation, and Alexander Werner from the Foundation's membership committee. This is the non-profit organisation at the heart of LibreOffice, the famous fork of OpenOffice.org now dominant in every Linux distribution. We were able to ask Florian about the split, about arguments over a new name and what wheat beer he'd recommend as a souvenir for our journey home.



The Document Foundation has a board of directors, which includes Michael Meeks works. What's your job as head of the board?
Florian Effenberger: I'm active in two parts. One is lots of Foundation management handling, like trade and operational tasks, going from tech staff, insurance, legal stuff, trademarks and all that goes with running such an entity. And the second part, which is the reason why I have Alex with me, is the infrastructure part. So I'm active in those two different areas right at the moment. The board runs the Foundation's daily operation to make sure it works, overviewing things and overseeing things, budgeting, and all the jobs like that.

Is it right that you did a lot of marketing for OpenOffice?
Florian: Indeed. It started about 10 years ago. I'd been an OpenOffice user for quite a while back then, and just by coincidence, I got into it. And when you don't say no fast enough, you can getsucked deeper inside. So I did quite a job of marketing with them.

What's the biggest challenge when marketing open source software?
Florian: I think the market has changed quite a lot over the last 10 years. When we started, it was basically that people were looking at you and asking questions like "Is it free?" and "How do you finance yourselves?" People were suspicious of free software.

People must think "What's the catch?", especially with something a big as an office suite!
Florian: Yes, it was always the same question, like "How do you coordinate yourselves?". And it was rather new. Linux on the desktop isn't extremely popular today, but back in 2004 there weren't many user-friendly Linux distributions as there are now. It was all quite uncommon.

That has changed. People accept you and expect you to be at trade shows and to have a photographer. We host our own conferences, also for professional audiences that had changed a lot over the last years. And so we have the challenges that we have to face. Like in the beginning, it was explaining what we are, how we do that, and we weren't so well structured. When you run your own conference, when you have a target market like the enterprise sector, you need to reach out and get them involved. The focus has not shifted, but has been widened so to speak. So the challenges change over time. I see similarities with other projects, facing the same issues.



> "When you have a target market like the enterprise sector, you need to reach out and get them involved"

Are there any groups that you really focus on? Like getting LibreOffice into schools or governments?

Florian: We are focused on everyone, because we have a rather wide target audience. What actually happens is governments are a rather large adopter of free software, so we obviously cover it a lot with them because they use it on a wide scale. So to give you one example, we have a large list of adopters in various countries. The occasional sector on the international level is, from what I can see, not so much represented at the moment. We are running a trial in Stuttgart, trying to gather some experience here. And otherwise, there are a couple of events we try to attend, but they mostly focus on the IT target audience.

> "Back in 2004 there weren't as many user-friendly Linux distributions as there are now"

Do larger groups still have some prejudices against open source?

Florian: Not so much. Every once in a while of course such discussions pop up but they are not really to be taken seriously. I think nowadays that LibreOffice is an established presence. And so, especially in the European market, with the creation of the foundation of the Stiftung [a non-prefit foundation], that's rather a sign of trustworthiness, because you don't just set up a Stiftung in five days. It really is a chunk of work and you really need to do professional work to set that up. It gives you quite a lot of credibility.

Now, instead of questions like "How can this be free" you have questions like "How do we do a migration?", and that's what we always try to tell people, "The software is free, you can use it free, you can edit it and all those freedoms, but if you want to deploy it on a large scale, you need some professional support." It's the same as for proprietary software, there's no difference. That's a message that you've probably seen in our press releases recently. We also work on a certification program from the TDF side. So by having a good ecosystem and professional partners, you are able to roll out large migrations and deployments. It's an important message at each stage I think, and not



LibreOffice has seen a huge increase in the number of contributors since the fork from OOo.

so much how good or bad free software is.

Have you seen a change in attitude since the NSA and Snowden leaks last year?
Florian: Yes, I think so. Looking at the press or at personal friends who are also not so much into IT, they are thinking twice about where to host their data and what to do, and who to give their data to.

That makes quite a difference. In terms of LibreOffice, I think that it's a message we have been spreading for quite a while. Like, you have open format, you have no Windows lock-in, so we've giving exactly the same message. Of course, it's been amplified these days by the concerns that have finally made it to the public.

Let's think about the current situation with Apache OpenOffice and LibreOffice. It seems uncomfortable to have two projects doing almost exactly the same thing. What's your take on it? Could they merge? LibreOffice seems to be getting much more attention, so what do you think will happen?
Florian: My take is that back in 2010 we had this hard decision to make: what should follow. Back then, there was no realistic chance to have ownership of the OOo project. A truly open, independent project. So we took a decision and, looking at the numbers of contributors and the feedback we got, I think it was the absolutely right thing to do and still is the right thing to do [to fork from OpenOffice]. If you look at today's market, what I can say from all the people I know in migrations when a new software release is due for roll out, like for the city of Munich, or when the first initial steps to a free office are taken, in nearly all cases it's LibreOffice. So I'd like to not so much talk about what others do, but rather the good things that we are doing.

Do you speak to the OpenOffice guys? Is anyone else talking about merging the two projects together?
Florian: I'm not aware of any talk of a merger. We of course have good contacts. They attend mostly the very same trade shows that we attend. We have good times, but we're always trying to focus on improving LibreOffice. What we always say is that our door is always open. Our project is really transparent and people can always contribute to it. And that is working quite well. Speculating about what could happen, or what would have happened, is rather hard.

Yes, but we just think about the effort that's being duplicated. It often makes it hard to spread the message when there are multiple products doing the same thing.
Florian: Yeah, it's not that easy to explain the story about what happened and why things have happened for people who aren't close to the project. And of course, for legal reasons, we had to come up with a new brand, so we came up with LibreOffice.

We're geting good feedback about the brand, especially here in Germany. People are very supportive of LibreOffice; we've seen the statement from the city of Munich, which in October 2012 said that it will migrate to LibreOffice with its next big IT rollout. For us, that's proof that we are doing the right thing.

Were any other names considered apart from LibreOffice?
Florian: Quite a chunk!

Can you give us some examples? We interviewed Richard Stallman a couple of years ago and he seems to regret using the term Free Software because it gets mixed up with shareware. He said, if he were creating the term now, he would use Libre Software or something.
Florian: This is reflected in our statute. We have this mission statement of the Foundation's objectives. I don't know it word-by-word, but it says we produce free open libre software, so we have all

of those three words in our statement. Of course, we had a hard time coming up with a names. The problem was back then that we didn't know how long they would last. There was a chance that we could work with the OpenOffice brand, and so we didn't know how the time we invested would work out.

There was quite a bit of negative feedback when you first forked.
Florian: Look at OpenOffice.org! We lived with the OpenOffice.org brand for 10 years and people were aware of that, and suddenly it had a different name! It was new – everything is new in the beginning, it's unfamiliar. I think that was to be expected, and it was only for a few weeks and then that was over.

Nowadays, it's a strong brand, quite recognisable. Whatever name you come up with, there will always be somebody saying that it is stupid or I can't pronounce it, or I don't like it. But in the end, we took a fair amount of time to come up with the brand, and for TDF we were sure we wanted to keep the name. We needed an entity that was short and even if we could have taken the OpenOffice.org brand, we wanted to have a different name for the overall entity. So that name was sure to be in on the long term. For the software, we didn't know whether we would be able to simply take the OpenOffice.org brand and move forward with that, so despite all the work invested in LibreOffice's name we weren't sure whether we would keep it, whether we would keep that. And as history tells, we stuck with LibreOffice, and we're quite happy.


LibreOffice has thrived since its fork from OpenOffice under the aegis of The Document Foundation.

In the beginning, I wasn't so happy with the LibreOffice name. At least, I recently backed up a load of data and read some comments that people made about the choice of name and I think it was me saying, "oh it's OK but I'm not totally happy", but nowadays I'm rather happy with the brand. It really needs to grow, and you need to get comfortable with it and familiar.

"We're getting good feedback about the LibreOffice brand, especially here in Germany."

Wasn't LibreOffice a short term name originally?
Florian: It wasn't short term. We wanted it to be there on the day we made the announcement, to prove that we were serious.


We loved Florian's LibreOffice stickers, but didn't get a decent photo. Here's some beer instead.

We had a big discussion about the name of our magazine. Half the challenge is just making the decision. Then once you've done that you have something to get behind, and we think that's what's happened with LibreOffice.
Florian: Indeed! Yes, we had a long list with nothing that had a majority. So we had about five or ten candidates that could have worked and in the end we voted, but it was a big decision.

A lot of people just don't like change though. We remember seeing a while back on Mac Rumours, that Apple had changed the icon for iTunes for OS X v10 and there were 600 comments in the thread. And people were even saying they would never buy another Apple product again! It was quite scary.
Florian: Absolutely, it needs to grow. If you get an agency to come up with a brand for you, it costs a fortune because they spend a considerable amount of time thinking and analysing. I think they just sit in a room around a table with a very large list of names, just like we did. I'm quite happy with our branding of LibreOffice. That reminds me: I've brought something for you, because I love LibreOffice! [Florian produces some LibreOffice stickers]

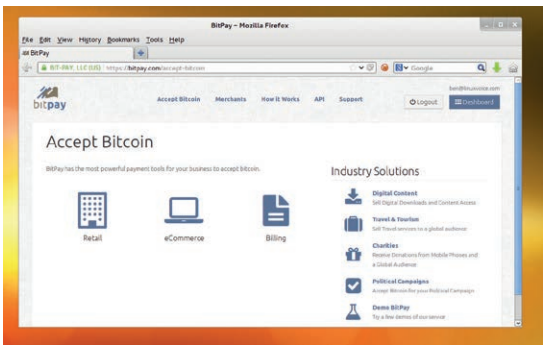
Everybody loves stickers!

bitcoin



What in the name of Zeus is it?

Bitcoin is a digital currency that became popular in 2013. It's not controlled by governments, banks, or anyone. It's a decentralised currency designed to free our money from those who would oppress us. But how does a digital currency work? How can it be valid if there's no one to say who has what?
Ben Everard investigates.



Sellers can receive money in Bitcoin without having to deal with the currency itself, using services such as BitPay (www.bitpay.com).

There are, roughly speaking, three parts to Bitcoin: the block chain, mining network, and wallets. In order to understand how Bitcoin works, you have to understand how each of these works. Make a cup of tea and settle in.

The block chain is a list of every single Bitcoin transaction that's ever taken place. Until a transaction is on the block chain, it hasn't happened. It is quite literally a chain of blocks – each block is a list of new transactions, and a link back to the previous block. Anyone can then validate the block chain by following it all the way back to the very first transaction when Satoshi Nakamoto created the first Bitcoins.

At this point, you're probably wondering who's responsible for keeping the block chain. The scary answer is: no one. There is no single organisation or person that holds a definitive copy of the block chain. Bitcoin is built to be distributed, so there's no point of failure that could maliciously or accidentally corrupt the block chain. Instead, the block chain is held separately by every single computer mining Bitcoins.

Where there's silicon, there's brass

These miners, then, are both the custodians of the old transactions, and the ones responsible for making sure new transactions are added. Their job is to create (or 'mine') new blocks. These new blocks contain any new transactions that have taken place. In compensation for mining these new blocks, they're rewarded with some Bitcoins. This acts as an incentive to make sure enough people keep mining to keep the network working.

That brings us to wallets. This is the part of Bitcoin that regular users see. The term wallet is a bit of a misnomer, since they don't actually store Bitcoins at all -- Bitcoins are stored only as a record of transactions in the block chain. The wallets store a private key that authorises the user to add transactions to the block chain for a given address (which is the public key that corresponds to the private key).

> **"The term 'wallet' is a bit of a misnomer, since they don't actually store Bitcoins at all."**

HASHTAG

Hashing – sometimes known as one-way encryption – is a method for changing something in one way that can't be reversed, but can be verified. Take for example a very simple hashing operator: modulo 10. In this, you divide something by ten and the remainder is the hash – for example, 45 hashes to 5.

There are two crucial functions about the hash. The first is that it's easy to verify. Every time you do it, it's quick and you get the same result. The second is that you can't reverse it. If someone tells you that the hash is 5, it's impossible to work out that they started with 45.

However, modulo 10 is a bad hashing algorithm because it's easy to find something else that hashes to the same value. A good hash has both of the first two properties, but make it impossible to predict how the output will changed from a change in the input. A slight change in the input should result in a drastic change in the output.

Hashes are used frequently in computer security. For example, it's how passwords are stored on Linux systems. The passwords themselves are never store; instead, their hashes are. You can see them if you type:

```
sudo cat /etc/shadow
```

Every time you log in, your computer hashes the password you type and compares the result to these stored hashes. If the two hashes match, then it logs you in. The fact that you can see every hash on the system (if you have superuser privileges) doesn't make this any less secure because it's so hard to reverse these hashes. In fact, it's only really possible if you can guess what the hashes might be (by checking with a list of dictionary words, for example).

In Bitcoin, hashes are used both to verify the integrity of the block chain, and in the proof-of-work (see the boxout below). They work in the block chain by proving that none of the blocks have been edited since they were first mined.

HASHCASH

Hashcash is the proof-of-work system that miners use to verify that they have actually mined a block before it can be included in the block chain. The basic function of this is to make it computationally unfeasible to alter the block chain, because anyone seeking to alter a transaction would have to recalculate all the proof of works until they had a block chain longer than the previous one.

It relies on hashing (see other boxout), specifically the SHA256 hash function. This takes an input and outputs a 256-bit number. The inputs to the hash function are the block header (which contains a counter) and a hash of all the transactions. The task of the miner is to find a value for the counter where the output of the hash function is below a certain threshold. This threshold corresponds to the current difficulty setting, which changes every 2016 blocks.

The only way to calculate this is with pure computing power. You have to generate as many hashes as possible with different values for the counter and hope that one of them comes up with a hash below the set value. The faster you can generate hashes, the more likely you are to find one that satisfies this requirement. If you do come across a hash like this, then you have mined that block and you can transmit it to all the other miners in the network.

The rate at which miners are trying different hashes is used to show the current speed of the network or the power of a particular Bitcoin-mining computer (usually measured in billions of hashes per second GHs)

Miners don't have to worry about their proof-of-work being copied because, it includes a hash of all the transactions (in something called a Merkle Tree), and one of these transactions is the miner paying themselves for mining the block. Anyone copying the proof-of-work can't alter this without changing the resulting hash.

Note that the hashcash algorithm used in Bitcoin is slightly different from the hashcash algorithm used to prevent email spam, though they both work in the same general way.

MAKING A TRANSACTION

When you make a Bitcoin transaction, you transmit it to the network of miners. However, there has to be some security to ensure that someone else can't make transactions from your wallet without you knowing.

Bitcoin transactions take place between two (or more) wallets. These, as we said before, are simply a public/private key pair and are used to encrypt data. They work in such a way that any data encrypted with the public key can be read with the private key, and vice versa.

In Bitcoin, you don't have a pool of money that goes up and down like a bank account. Instead, you have a specific set of Bitcoins that can each be chained all the way back to their original miner. When you make a transaction, you have to reference the transaction in which you got them (you can reference more than one). You then have to digitally sign each referenced transaction. This means that you hash the details of the transaction and encrypt them with your private key. Since your public key is tied to your address (and therefore tied to the referenced transaction), this confirms that you are authorised to make the transaction. The transaction also includes the output address to which they are being sent.

(This is a bit of a simplification. See **https://en.bitcoin.it/wiki/Transactions** for a more complete explanation.)

Notice that nothing physically leaves your wallet other than this message to the block chain. The amount of Bitcoins in a wallet is calculated by seeing all the transactions in the block chain. This means that anyone can know how much is in any wallet at any one time. It doesn't necessarily mean they know who has how much money since it's not always possible to tie a specific wallet to a specific person, or know how many wallets a person has.

Once this transaction goes to the miners, it's added to the next block. However, as we've seen, the block chain can split (and a malicious miner with a lot of computing power could split it deliberately). There is no one point when it's guaranteed to always be in the block chain, but the assurance is calculated by the number of blocks built on top of it. If it's just one, then a lucky attacker may be able to outrace the rest of the mining network. However, with each subsequent block that's added to the block chain, the amount of work an attacker would have to do to reverse the transaction increases.

A depth of six blocks is usually considered enough to be sure that a transaction is properly added to the block chain. At a block rate of one every ten minutes, this is an hour. For high-value transactions, you may wish to wait for more blocks before considering the money truly transferred.

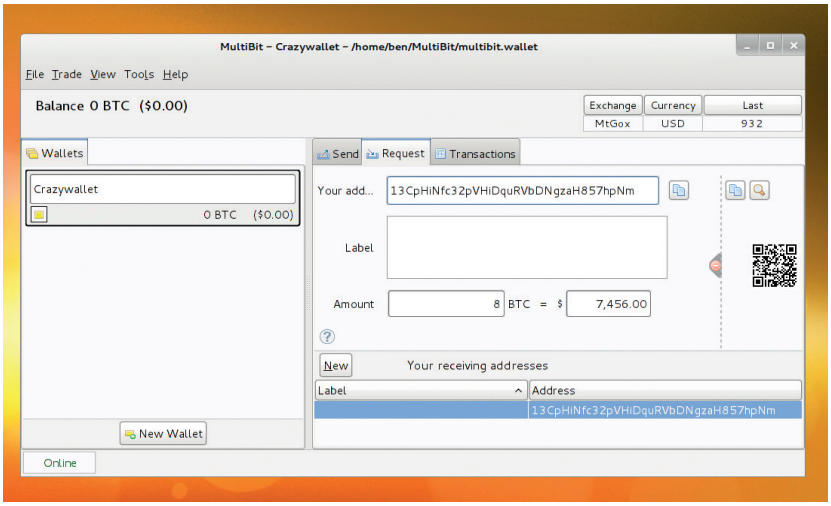
The previous four paragraphs have given a basic overview of how Bitcoins work, and you could quite easily go about using or mining Bitcoins using only this knowledge. However, the chances are that you wouldn't trust the currency, since it sounds suspiciously like it would be easy somehow to corrupt the system and defraud users. The beauty of the currency is in the cryptographic techniques that protect users. Let's go back and look at in more detail to see how this works.

> **"The beauty of Bitcoin is in the cryptographic techniques that protect the users."**

A huge part of the security of Bitcoin comes from hashing, and it's these hashes that are used to link blocks together in the block chain. Each block contains a hash of the previous block, and this can't be changed without altering the current block's hash (which will be included in the next block). Anyone can go and check that none of the transactions have been changed at any point. If they had, the hashes would no longer match up.

The block chain is a publicly verifiable record of every transaction that's ever taken place. Each time you perform a transaction, details of that transaction are propagated to all the miners on the bitcoin network with a request to include it in the next block.

For a miner to get paid for mining a block two things have to happen: they have to solve the



Multibit has all the usual wallet functions and can create QR codes to help someone send you money. These include the receiving wallet address and the number of Bitcoins.

SATOSHI NAKAMOTO

Bitcoin has become a household name, and the currency is worth billions of pounds, but for all that fame, one thing still remains secret: the identity of the creator. They're known only by the pseudonym Satoshi Nakamoto. It's not known for sure if Satoshi is male or female, or even if they're a single person or a group.

Satoshi mined many of the early Bitcoins (possibly up to a million), so at today's exchange rate, he (or she, or they) is a very wealthy person. It's possible that they have hundreds of millions of pounds worth of Bitcoins, but they have never spent any of them. Herein lies a problem. Since there's a strong suspicion surrounding which addresses are Satoshi's, if they ever spend any of their Bitcoins, they'll reveal their identity. So far, they've preferred to keep their anonymity rather than cash in; but will this last?

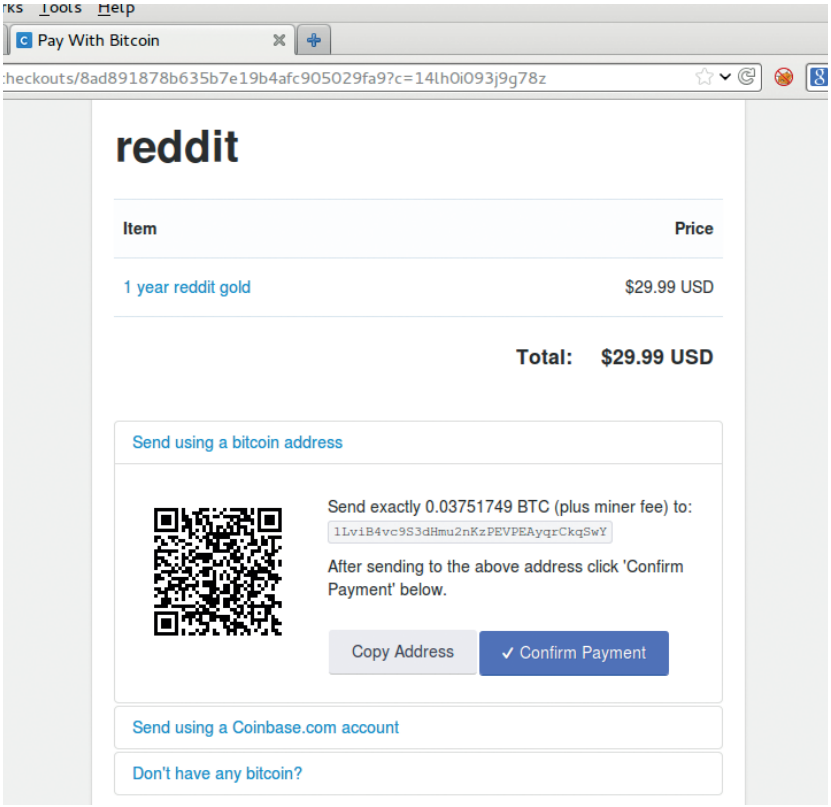
Of course, there's been wild speculation about who could have created the currency, and several researchers and journalists have pointed the finger at various people, but all have denied it. Perhaps at some point in the future, the lure of money will be too strong and the mysterious creator will reveal themselves. We can only wait and see…

ANONYMITY

Bitcoin has become the currency of choice for anonymous transactions online, most famously for illegal shops like Silk Road. However, it isn't designed as an anonymous currency, and in fact doesn't really fulfil the role particularly well because of the block chain. This records every single Bitcoin transaction and allows anyone to view the exact path that any Bitcoin has ever taken.

The only mitigating factor in the public display of information is the fact that you can create a Bitcoin wallet without telling anybody who you are. In this sense, the wallets are private, but the currency is completely public. This means that if you get Bitcoins anonymously (for example, if you mine them, or buy them with cash in an untraceable way), spend them in an untraceable way (for example, paying for something that's not delivered or tied to you personally in any way), and you don't link the wallet to your physical location (for example, only connect it to the internet through Tor), then the transaction will probably be anonymous. However, that's an awful lot that has to go right. If you slip up in any one of these ares, then the transaction is probably traceable back to you.

It is possible to make an anonymous transaction with Bitcoin, but, in most cases it'd be far easier to stay anonymous with cash. The block chain is a big-data analyst's gold mine and will almost certainly be used by law enforcement more and more in the future.



A typical screen to pay by Bitcoin. It contains all the information you need to make the transfer.

hashcash proof of work, and that block has to be included in the block chain. The first is purely a technical challenge, but the second is what forces them to check everything. If the block contains invalid transactions (for example, if someone is spending coins they don't have), and they do the proof-of-work, when they send the completed block to other miners, the other miners will reject it. This means that the original miner doesn't get paid, and has wasted his time. Therefore they will check every transaction to make sure it's valid before including it in a block.

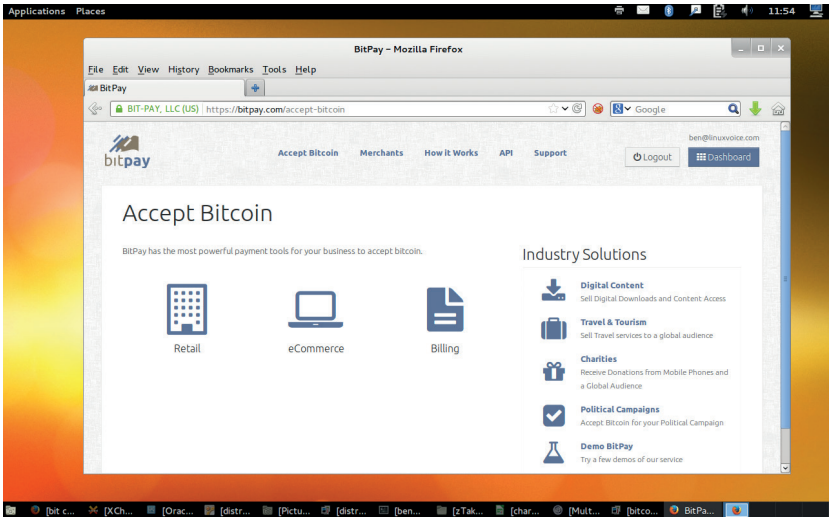
Honest self-interest

When a miner receives a block from another miner, they have an incentive to try and find fault with it for two reasons. Firstly, if they can reject the block, it means that they are still in with a chance of mining it themselves. Secondly if they accept a block that other miners reject as invalid (if they don't properly check it, for example), then any mining they do that builds upon that block will be wasted because it will never main it into the main block chain.

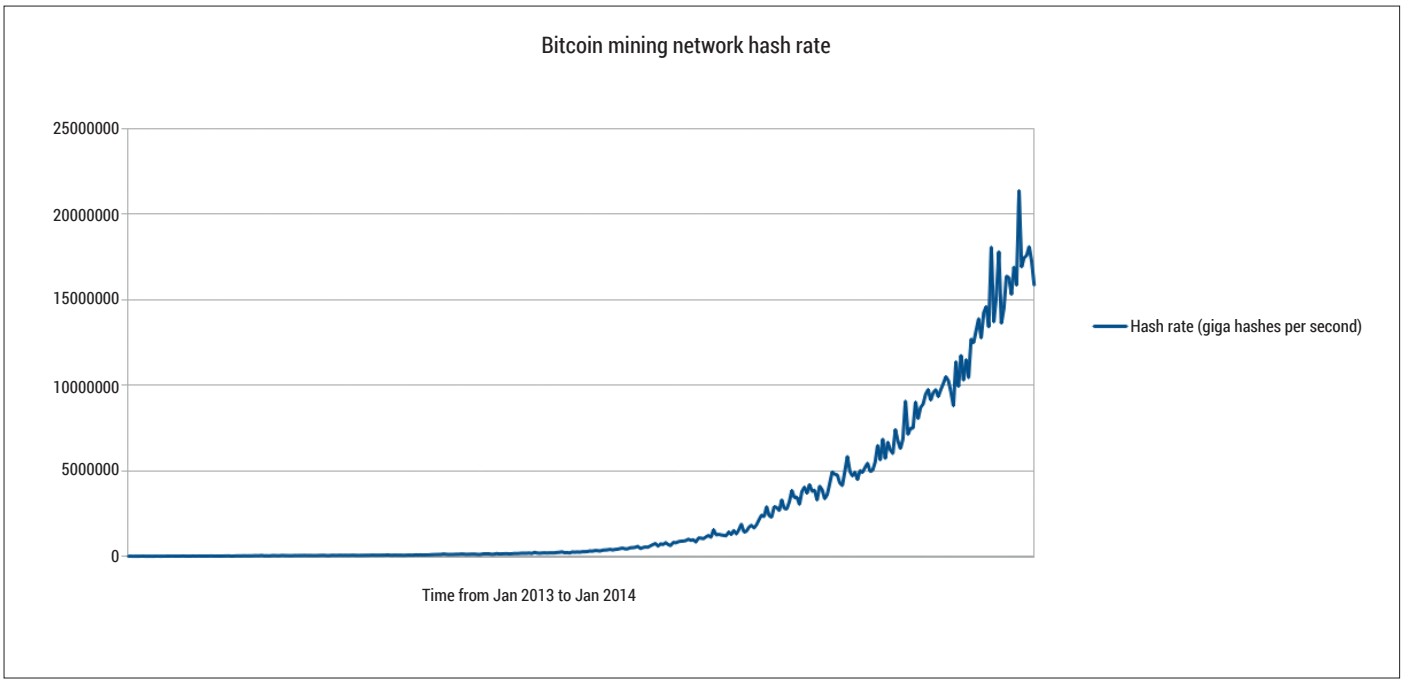
At the same time they have an incentive to accept valid blocks, because if they reject a block that everyone else accepts, then any subsequent blocks they mine will be rejected by the rest of the miners.

There is a slight issue that two miners could generate the same block at roughly the same time and send it out to all the other miners. At this point, there would effectively be a split in the block chain. Some miners would work on one, and some on the other. The rules of Bitcoin say that the longest valid block chain is the right one. One of the two splits would be first to get the next block, so then the bitcoin miners would move over to that because anyone mining in the shorter chain isn't going to get paid for mining unless it somehow overtakes the longer one, which will be increasingly unlikely as the longer one gains more and more miners.

These rules ensure that a network of miners who are each out to maximise their own profit will keep the integrity of the currency. A group of malicious miners seeking to somehow undermine the system need to have more computing power than all the legitimate miners combined (so they can mine blocks at a faster rate and maintain the longest block chain). The



You can buy Bitcoins from exchanges like this one (www.mtgox.com). They don't all have the same price or reliability, so it may pay to shop around.



The strength of a cryptocurrency is in its mining power. As you can see, this has increased dramatically in recent months.

hashcash proof-of-work, then, protects the Bitcoin network through raw computing power.

This is the reason Bitcoin mining has to remain profitable. As it currently stands, the Bitcoin network is performing at about 15 peta hashes per second and rising fast (see above). To buy computing power to beat this (if you bought the latest mining machines, although this would be impossible since there isn't enough of them to do this many hashes), it would cost around £150M (based on good Bitcoin-specific hardware). That's just for the hardware to equal the mining pool at the time of writing. The power of the mining pool doubled in the last month, and it's still increasing quickly. This also doesn't take into account electricity (they use alot), storage, cooling, people to run them, etc. Realistically, any attack on the block chain would have to perform so much fraud to cover its costs, that Bitcoin would crash in value, and the attack wouldn't be worth it.

Fraud prevention

However, the above only stays true while it's sufficiently profitable to mine bitcoins. Should this change in the future, then miners will take their computing power elsewhere (or stop upgrading it). This will then lead to a situation where the network could be exploited.

The profitability of mining Bitcoins is controlled by two factors: the difficulty in mining each block, and the number of Bitcoins the miner gets for each block.

USING BITCOIN

The crux of a currency, for most people, is how you use it. For most real-world currencies, this involves handing over metal discs or pieces of paper, but there are no such things for Bitcoin.

The first thing you need is a Bitcoin wallet. This is really just a public/secret key pair that's used to sign transactions. However, you need somewhere safe to store this because if you lose it, you can never get the coins back. How safe depends on how much money you want to store. There are wallets for almost all computing platforms, including smartphones. Remember that there's no cost to setting up a wallet, so there's nothing to stop you from having several. For example, you could store some of it on your main computer and some on your phone, meaning that you can spend it on the go. It's also possible to get wallets that are hosted online, for example, at **blockchain.info**. However remember that with online wallets, the host potentially has access to your Bitcoins.

All Bitcoin wallets work in roughly the same way, and have all the information you need to send and receive Bitcoins, and to see previous transactions.

After creating a wallet, you need to get some coins. For most people, this means buying them from an exchange such as **www.mtgox.com**. Unfortunately, this isn't as straightforward as buying most goods, since it's not usually possible to buy them with a credit card or PayPal. This is because fraudsters have previously made money by buying via these methods, then complaining to the card company that they never got the coins, and reversing the charge leaving the seller out of pocket. The credit card companies could easily check the block chain, but in the past they've chosen to side with the purchaser and now none of the major exchanges accept card. The result is that generally you have to pay by bank transfer. As it stands, this is probably the biggest thing putting most people off buying Bitcoins [*It put Ben off buying some for this feature, and in the last few days the price of Bitcoins has increased by 50%. Oh well, never mind*].

When you buy Bitcoins, you'll be asked to provide a wallet address. Once the bank transfer has taken place, the exchange will transfer the coins to the specified wallet. You then own the coins and can transfer them to whoever you wish.

Spending coins is far easier than buying them. More and more companies are accepting Bitcoins every day. When you go to a checkout, you'll be given a wallet address to transfer the funds to (this is usually a wallet created just for this transaction so don't try to reuse the same address in the future). This is often expressed as a QR code. If you have a phone wallet, you can usually transfer the funds just by taking a picture of this code. Remember that you don't send the transaction to the receiver, but rather to the mining network. The person receiving the money then gets the block chain from the mining network and looks for a transfer into the appropriate wallet.

The company receiving the funds will usually wait until the transaction has reached a depth of six or more blocks, which could take around an hour. Once this is done, you should receive the product. Remember that bitcoins are like cash, and there's no way of getting them back should the person you pay not supply you with the product.

These two things also have to be balanced to account for rising computing power and the rising market value of Bitcoins.

The number of Bitcoins awarded per block changes at a fixed rate: it started at 50 and halves every 210,000 blocks (approximately four years) until 21 million Bitcoins have been mined, then no more are awarded for mining subsequent blocks.

The difficulty is varied every 2,016 blocks. The network is designed to generate a new block every 10 minutes on average. This time was picked to be a happy medium between two opposite forces: shorter times would make transactions happen more quickly, but are more likely to lead to more than one miner generating a block at the same time, which leads to wasted resources as the two block chains compete to become longer. To keep this time period despite the hash rate of the network varying wildly, the algorithm looks at the time it took to generate the previous 2,016 blocks and tries to compensate for this.

You may have noticed a slight flaw in the plan. First we said that the security of the system was dependent on the amount of computing power it had available which was in turn dependent on the profitability of mining. Yet then we said that at some point in the future, when 21 million bitcoins have been mined, there will be no more rewards for mining.

This isn't quite true. There will be no more new bitcoin rewards for mining, but when you carry out a transaction you can include a transaction fee, which goes to the miner. At the moment this is rarely done, since mining is profitable enough that people do it without this additional incentive, and the transaction volumes are low enough. However, each block is limited in size to 1MB (there is currently a debate about whether to change this). This means there's a fundamental limit on the number of transactions in each block. If a miner reaches a situation where there are more transactions than there is space in the block, they have to decide which ones to put in. Obviously the miner will go with the ones that have the highest transaction fees. The other transactions won't be lost, they'll just be rolled forward into future blocks. Higher transaction fees will result in faster transactions.

At the moment, most Bitcoin transactions have no transaction charge, but this isn't a fundamental feature of the currency that will stay with it forever. It remains to be seen whether the transaction fees will be less or more than for other payment methods in the future. I really wish I'd bought some when I started writing this feature!

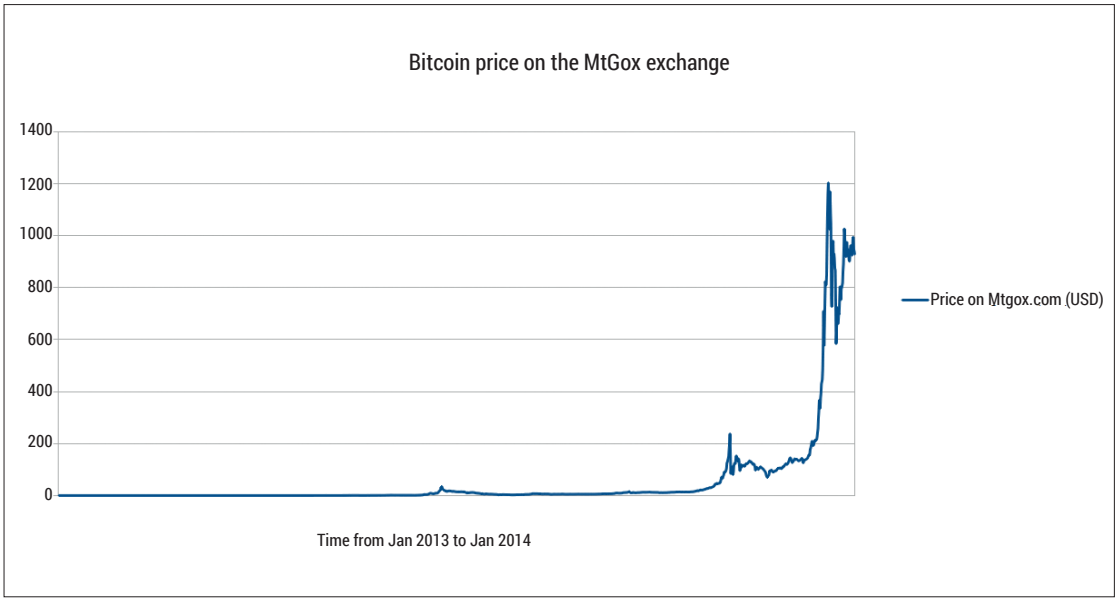
"When you carry out a transaction you can include a fee, which goes to the miner."

WHAT IS MONEY?

One of the most common concerns with Bitcoins is that they aren't real money, just numbers on a computer. This, in a sense, is true, but what actually is money? Once upon a time it was linked to physical goods, and some of the currency names reflect this (A British Pound was originally the value of a pound of silver). That link between currencies and physical goods, though, is long since gone. The last major currency to lose the link was the US Dollar, which was tied to gold until 1971. Nowadays, currencies don't have any value other than what people place in them, and the varying values people place in them is what makes exchange rates change over time.

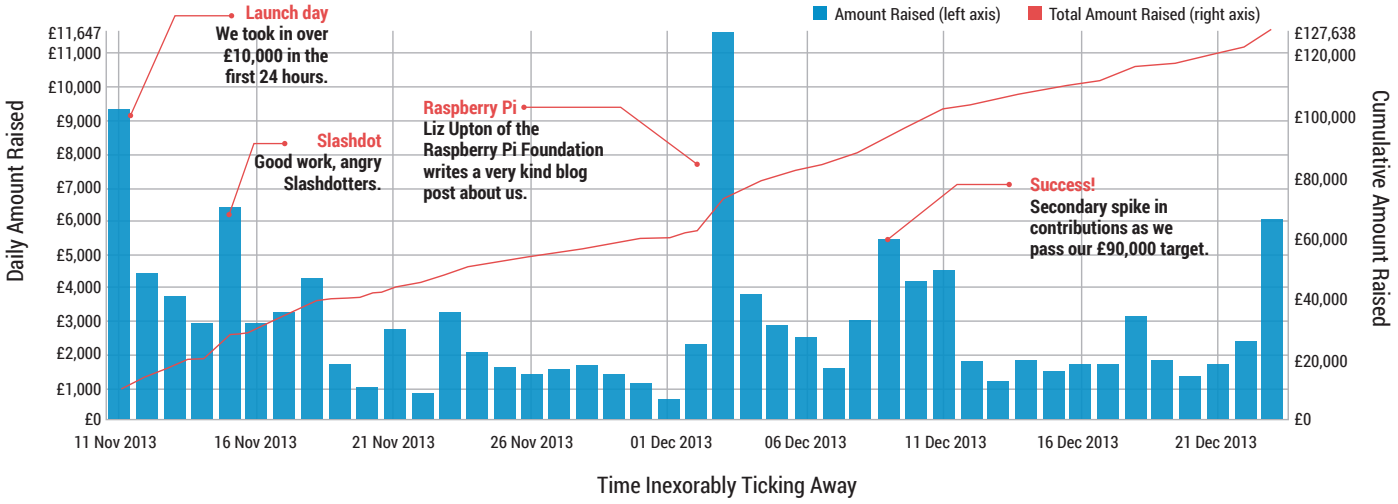
The only real difference between Bitcoin and a national currency is that national currencies are backed by governments, whereas Bitcoins aren't really backed by anyone other than the miners. Whether or not this is a good thing depends entirely on your economic philosophy. On the positive side, no one can print excessive amounts of money leading to excessive inflation (like the German government did following World War One). On the negative side, there is no one to step in and help stabilise it should things start to go wrong (for example, the various governments that printed more money to help ease the cash flow crisis in the 'Credit Crunch').

In practical terms, the biggest difference between Bitcoin and most other currencies at the moment is the wildly fluctuating exchange rate. The value of the currency can double or halve in just a few days. Proponents say that this is likely to stabilise as the currency become more popular, and this is probably true, at least to some extent. In reality, though, no one knows what will happen in the long run since currencies without the backing or a state or organisation haven't been tried before.



The dollar value of Bitcoins has increased by almost a factor of 100 over the last year. Some people see this as a bubble set to burst, others as a sign of the currency coming of age.

The total amount Linux Voice raised (yellow), and the income by day (blue). We tracked these blue and yellow lines hour by hour as the project progressed and learned to love and hate them in equal measure.



The story so far

This magazine crowdfunded its way into existence. **Ben Everard** tells the story of how it happened.

As long as there have been jobs, there have been people sitting in pubs complaining about them. The Linux Format team had been meeting in The Salamander in Bath to do just that since long before I joined. It was a safe place that served good ale, and it provided a safe, warm place to moan over a pint. But there's only so much moaning you can do before you have to quit, which, being men of action, is what we did.

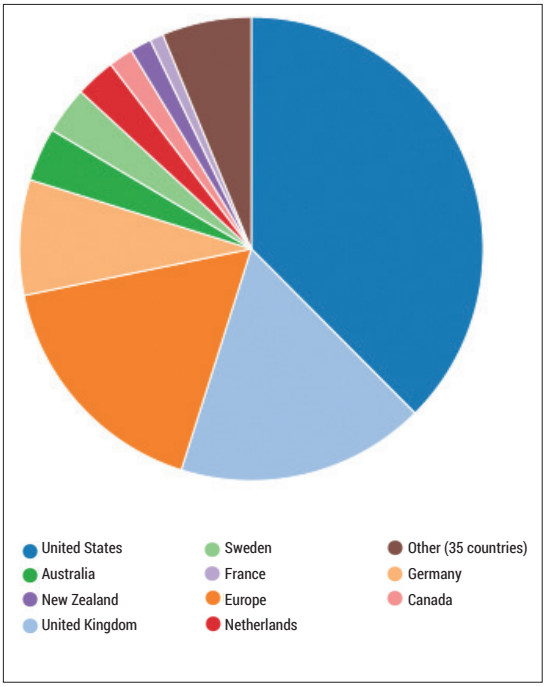
But something carried on. We were all used to making a podcast together, so we continued; or rather, we started a new one, called Linux Lifestyle. Yes, it's a rubbish name, but the domain was cheap. After one recording, back in the Salamander, we had an epiphany: if we could make a podcast together, we could make a magazine together. Not all ideas from the pub seem so good in the cold light of day, but this one did. Only one thing stood in our way: the lack of money. We needed a plan.

At midday on Monday 11 November 2013, linuxvoice.com went live announcing to the world our intention to create a new magazine using Indiegogo to crowdfund the launch. For us to be successful, we needed to raise at least £90,000 before 23 December. If we didn't hit that amount, then there'd be no magazine and we'd all have to start looking for new jobs in the new year.

The floodgates open

After what seemed like an eternal wait before we went live, the subscriptions started coming in. One hour after we went live, the total had reached £1,525. Not bad for an hour, but it still barely registered on the progress bar.

There are a lot of ways you can analyse the money coming in. The simplest way is to say that we needed to raise £90,000 in 43 days, therefore we needed to get a little over £2,000 each day. Looking at it this way,



A breakdown of our income by country. Thanks America!

we'd almost reached our daily target in just one hour. In fact, if things kept going like that, we'd raise over £1.5M! Of course, that's an oversimplification. What if we look at it another way? All the Linux Lifestyle podcast listeners would have just received an update in their RSS readers telling them what was going on. Pretty soon, that would get buried under newer items. What if this is now everyone who cares? What if the rest of the world is perfectly happy with the status quo? What if there simply isn't a demand for a new magazine? What if the rumours are true and the magazine industry really is dead?

For most of the remainder of the campaign, my mental state fluctuated between euphoria and despair as I considered the good and bad possibilities. There was no 'OK' outcome, so it was all or nothing. At first though, my goal was much more modest. I simply didn't want to be humiliated. Failing is one thing, but if we barely raised anything. It would be on the internet for all the world to see. All my friends and family were checking the site. If the progress bar never moved, and stayed firmly in the red, it'd be one very big, very public humiliation.

In truth, I don't remember much more of that day except bouts of fear and happiness. Looking back, I can see that we sent out press releases and started to get active on social media, but all I member is a quickened pulse and an abject fear of public humiliation. It took quite a lot of alcohol to calm me down enough to sleep that night.

I woke at sunrise the following day. I didn't even turn the light on, I just reached for my phone and hit refresh (it still had the crowdfunding page loaded from the night before). We'd topped £9,000! 10% of our way there in just one day, and we'd almost hit five figures! Whatever else happened, it wouldn't be a humiliation.

> **"All I remember is a quickened pulse and an abject fear of public humiliation."**

In hindsight, it probably should have been obvious at this point that everything would work out, but it wasn't. I still struggled with the constant fear that we'd run out of supporters. It would be another three weeks until this finally abated and I believed it would happen.

Getting heard

Success in crowdfunding comes down to many things. You have to have a product that people love enough to take a chance on. We thought we had this because we'd had so much good feedback working on our previous magazine, and we had some really excellent ways we could make *Linux Voice* better. However, it doesn't matter how good the product is if no one hears about it. From day one onwards, we devoted ourselves almost entirely to publicity.

The majority of the publicity came from social media (though we did feature briefly on the Guardian

WHAT IS CROWDFUNDING?

If you've picked up this magazine in a newsagent, you may not realise that we launched it through crowdfunding.

Crowdfunding is a way of raising money for a product or project by getting people who are interested in the project to put money towards it. In return, these funders get 'perks' which can be anything from a word of thanks (see our Founders page) to a copy of the product, to anything else. In a sense, this is a bit like pre-selling a product before you produce it, but for legal reasons it's a bit different because there's a chance that the product will never get made.

The two most popular websites for crowdfunding are Kickstarter (**www.kickstarter.com**) and Indiegogo (**www.indiegogo.com**). These sites hold thousands of projects looking for supporters. Some, like the Pebble watch, have raised millions of pounds.

This method of raising money has become particularly popular with computer game makers and graphic novel artists who can pitch their ideas directly to their fans rather than having to go through publishers, who in turn have to guess what the audience will and won't want.

TOP TIPS FOR CROWDFUNDING

If you've got a project that you want to start, but not the money to start it, crowdfunding can be a great way of getting off the ground. However, it doesn't work for everyone. Here are our top tips for a successful project:

- Give people a reason to love your product. If you have a product ready to ship, then it's enough to be a product that people like, but when you're asking people to take a risk, you need them to love the product.
- Understand who the influential people are in your field and target them. People are far more likely to trust them saying nice things about your product.
- The internet is already starting to tire of crowdfunding, and many social sites are pushing back against it. For example, the Technology subreddit has an outright ban on crowdfunding stories. To get past this fatigue, your product has to be exciting.
- Don't rely on mainstream media. Maybe you'll be picked up by newspapers and television, but it's more likely that you won't. Have a plan in place to reach your intended audience.
- Leave no stone unturned when it comes to publicity, and don't stop plugging away at everything.
- Videos are far more convincing than text, even when they say the same thing. It conveys a sense of personality that's easily lost in the written word.

website and The Register). We quickly learned that each of these works in a slightly different way. Hacker News is a brutal, fast-paced all-or-nothing ride. If your article makes it to the front page (as a few of ours did), you get a big hit of visitors, and a lot of comments that need answering very quickly. Keeping on top of a popular comment thread on Hacker News basically requires constantly typing as fast as you can for as long as it stays popular.

Reddit, on the other hand, also commands a big following, but it's spread over a much longer time (at least it is on **/r/linux/**). There are also lots of plenty of insightful questions, but they don't all come at once. Twitter is different because it goes both ways. We could seek out people and people could seek us out. We never found Facebook worked for us at all.

> **"The Linux podcasts supported us with an enthusiasm that no other media had."**

It wasn't all about new-fangled social media though. Philip Newborough (aka Corenominal) the creator of the CrunchBang distro, put a banner advert on its website supporting us. I regularly searched Google to look for new mentions of Linux Voice and found messages of support on forums across the internet.

As the first week went on the money kept coming in, but we got a bit less every day. From over £9,000 on Monday, it went to £4,436 on Tuesday, £3,775 on Wednesday and £2,959 on Thursday. We expected it to dip as the initial buzz wore off, but it was dropping away fast. If it kept going at this rate, it would die out completely by the middle of week two. On Friday, my mental state continued to oscillate between joy (when I realised that in just four days we'd raised almost a quarter of the total) and despair (when I looked at how much the cashflow was slowing down). By the middle of the day, I was spending more and more time in despair as I furiously hit F5 on the Indiegogo page waiting for the next donor. Each time the gap to the next one got a little longer.

Breakthrough

Then, in the space of a couple of hours, everything changed. First, Tim O'Reilly tweeted "Linux Voice: A new Free Software and Linux magazine that gives profits back to the community. Its @indiegogo campaign http://bit.ly/17XxdUg" to his 1.75 million Twitter followers. Two hours later, Linux Voice ended up on the front page of Slashdot. Suddenly the cash trickle became a cash flood. In the peak hour that evening we raised £1,680 – the best hour to date.

The feeling of sitting at your computer, pressing F5 and watching you get closer and closer to your dream is like flying. You've done the enormous amount of hard work to get airborne, then everything just comes together and you cruise. You glide along and the whole world is before you. It feels like anything's possible. It's peaceful, yet produces a surge of adrenaline. A giddy, almost child-like excitement takes over and your cheeks ache from grinning so hard. Also like flying, it doesn't last. We raised £6,399 on Friday, but less than half this on Saturday. What was worse is that we felt we'd exhausted social media and needed another source of potential new subscribers.

Despite being a member of a Linux podcast, I hadn't realised just how popular they were. Perhaps there's

ANDREW SAYS...



The thing that really hit me was that many contributors were paying, say, £55 for a UK sub rather than £50. Every time it happened it felt as thought the internet was giving us a silent nod of approval; that there were people out there who didn't just want the commercial exchange of getting a great magazine for a good price, but who wanted us to succeed for some other reason. Every time my phone went off with an email from PayPal telling me that someone had chipped in an extra pound or two, it felt like we were doing the right thing.

The low point came from a necessary evil: PayPal. On 15 December some international money laundering switch triggered at PayPal, which stopped it taking payments, which in turn convinced IndieGogo to shut down the campaign. We were stuck up a creek without a paddle, and even though it only took a couple of hours to fix the problem, the fact that it happened at all was a huge fail; the fact that nobody at PayPal or IndieGogo had got in touch to let us know what was going on, inexcusable.

We have so many milestones left to come. The first time we sponsor a project, the first articles we release as CC-BY-SA, the first time we see someone else building on our work, are all still to come, and it's going to be fantastic to see how the Linux Voice community grows. I can't wait.

GRAHAM SAYS...



Launching a magazine through a crowdfunding campaign must be a little like crowd surfing blindfolded. You stand at the edge of the stage, interpreting the sound of the crowd in one way but not knowing whether enough people are interested, or even listening. It's not until you throw yourself from the edge that you have some idea. But even then you don't know how far the crowd can take you, or what the landing will be like.

Unlike Ben, Mike and Andrew, I had a non-compete clause in the contract I had with our former employer. As I couldn't be involved in the campaign, I had something of a different experience to the other three. I had assumed that it would leave me with lots of free time for catching up on projects I'd neglected for years. In the end, those weeks of the campaign were probably some of the most unproductive of my life! I couldn't concentrate on anything, or plan anything, or write anything, or play anything. All I could do was watch the campaign page and bore friends and family with theories on how it might work out. They'd always nod, dutifully. However, I genuinely believed in the idea – a magazine with great content that gives back what it can to the community. Part of the growing distraction over those weeks wasn't doubt, but excitement at the prospect that Linux Voice might really happen. And it has!

MIKE SAYS...



Talk about a rollercoaster ride... At the start of the campaign, we thought we might be mad for even trying something like this. What if we only raised £5,000 of our £90,000 target? Will we become the butt of jokes for Linux journalists around the world? Will we all have to start new lives as goat farmers in Tajikistan?

But no, it worked. And it worked spectacularly, because the Linux and Free Software communities really understood what we were doing. We weren't just making yet another magazine – we wanted to do something different, away from the constraints of big businesses that didn't understand our audience.

Linux Voice has been a very personal project for me, because it's exactly the sort of magazine that I've always wanted to create. I've been reading computer magazines for the last 25 years, from the ZX Spectrum through the Amiga to the PC, and one thing has always struck me about the Linux community: it is incredibly passionate.

Sometimes there are arguments, and sometimes there are fallings-out. But we're all on the same journey, trying to make computing more open, free, exciting and fun, and a truly great Linux magazine should reflect that. You, the readers will be a key component of Linux Voice – not just consumers. We're but a tiny acorn right now and we have a lot to do, but thanks to everyone who supported us for making this dream a reality.

some form of brotherhood of the microphone that I'm unaware of, but the Linux podcasts supported us with an enthusiasm that no other media had. We made guest appearances and were featured many, including the Ubuntu UK Podcast, Linux Outlaws, The Linux Action Show Unplugged, The Linux Link Tech Show, Hacker Public Radio and TuxJam.

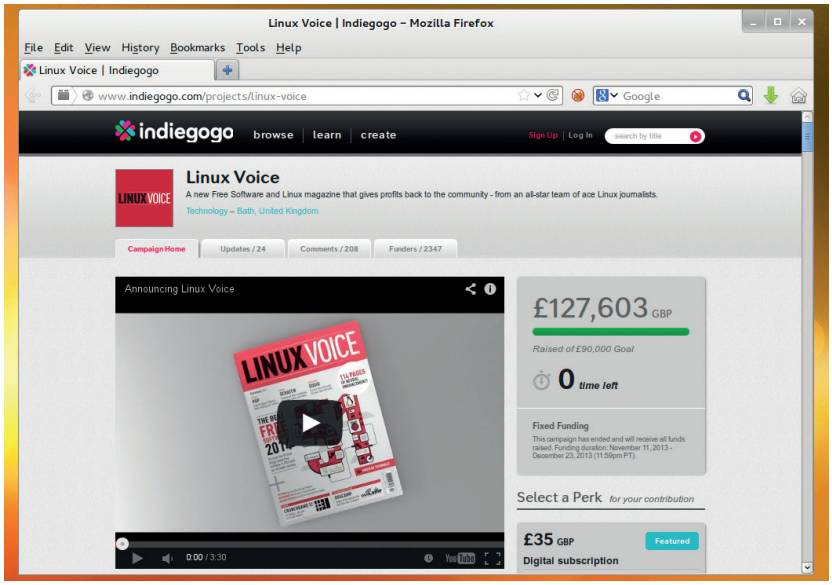
Each time one of these shows aired, we saw a spike in subscriptions on the site. Without the support of these guys, it would have been a lot harder. Not just because they helped get the word out, but because hearing other people within the community get excited about the project was a huge psychological support for me.

Bridging the gap

In crowdfunding, anything can happen, but we knew that most campaigns had a 'U' shaped cash flow graph. That is, they raised quite a bit at the start, and quite a bit at the end, but not that much in the middle. This middle section is known as the slump.

By week three we were well and truly in the slump. Income hovered around £1,500 a day, which was precariously close to the daily average we needed to hit in order to make our target. Of course, the theory said that we'd get a massive increase at the end, but would we really? There was no way of knowing.

Around this time, when I spoke to people, they'd have a pitying expression on their face if the subject of Linux Voice came up, and I could tell that few of my friends and family thought we'd reach the target at this stage (though they only admitted it later). I always



The campaign website got over 100,000 views during the campaign, which means one in fifty people that saw the page contributed. Thanks everybody!

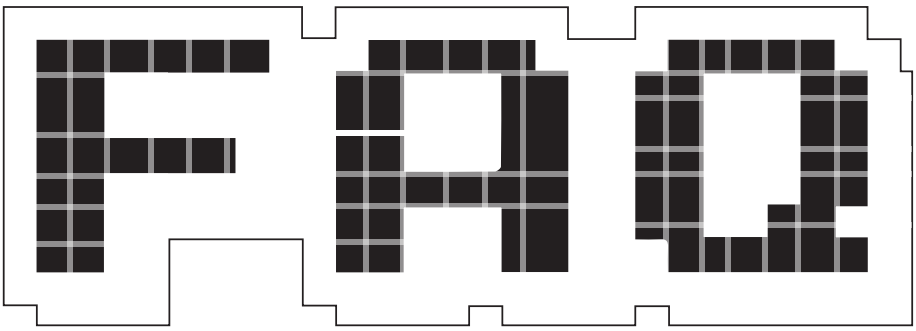
put on a brave face and told them of the slump and how every project hit it, and how people held out for when the deadline approached. As the week wore on, I struggled to believe this myself.

On the 1 December, just £649 came in. Was this it, I wondered? Had we finally run out of people interested in a new Linux magazine? For hours I sat pressing F5 and nothing would change. Then maybe a digital subscription, then nothing for hours again. I stopped alternating between hope and despair and stuck almost continuously in despair. This, I became sure, was us running out of supporters.

Crowdfunding, though, is a fickle mistress, and the best piece of advice I could give to aspirant crowd funders is to prepare for the unexpected. Two days after this low point, Liz Upton of the Raspberry Pi Foundation wrote a blog post supporting the project. This was picked up by Twitter, Hacker News and Reddit, and drove support faster than we'd seen so far, faster than we could have possibly dreamed. It peaked at £4,405 in a single hour (almost triple the previous fastest rate) and at the end of the day, we'd raised another £11,647. This pulled the total to £72,903. We now needed less than £20,000 with 20 days to go.

My girlfriend regularly asked me if I thought we'd make it. That night I replied with an unqualified 'Yes' for the first time.

The rest, as they say, is history. Following the endorsement from the guys at Raspberry Pi, the money kept coming in. Not at the same rate, but we made the final £18,000 in six days, and the total kept rising until the campaign ended with £127,603. The fruit of the campaign is now in your hands, and all of us at Linux Voice would like to say a hearty 'thank you' to everyone who supported us. Even those of you who waited until the last minute and made us sweat.



WAYLAND

The Anglo-Saxon god powering your next generation graphical desktop.

GRAHAM MORRISON

Q Why does Wayland need two pages of explanation?

A For most of us, Wayland is difficult to understand because there's nothing tangible to click on. There's no 'About' box to open or configuration panel to play with. But all of these graphical elements can be displayed on your Linux desktop using Wayland – it's just that Wayland hides beneath the surface. The important point is that it's a big improvement over the way this was done before, and is currently done now.

In simple terms, Wayland has the potential to make your desktops talk to your graphics hardware much more efficiently. Developers won't have to work with an arcane system that's massively over-engineered and complicated, while users should see performance benefits and more eye candy. It really could revolutionise the Linux desktop.

Q That sounds promising. What does Wayland actually do?

> **"More often than not it will be toolkits such as GTK or Qt that need to talk to Wayland"**

A Imagine it's the 80s. You're sitting in your bedroom prodding away at a Commodore 64, listening to the Human League and staring through thick-rimmed glasses at your 14-inch colour television. You're coding a game and working on the graphics. To get the best performance, your code is talking directly to the graphics hardware, the venerable VIC-II chip. One of VIC's best features was its ability to allow the programmer to create a simple graphical element, perhaps a spaceship or a gold miner, called a sprite. All the programmer had to do was tell a sprite what to look like, what colour to be and where to appear. They helped the programmer forget about the nuts and bolts of how their computer worked and concentrate on the gameplay.

This is what Wayland helps to do for the modern programmer. They can forget about the nuts and bolts of graphics and concentrate on usability. But because software stacks are now several layers deep, Wayland isn't aimed at the application programmer most analogous to our 80s games programmer. More often than not, it will be toolkits such as Gnome's GTK or KDE's Qt that need to talk to Wayland, and it's these that need to be updated to accommodate its requirements. Application developers shouldn't need to change their code, unless they're using something specific to the way the current system works. As long as the API supports Wayland, the applications will support Wayland and automatically look and feel awesome.

Q Brilliant! So how come it hasn't been adopted already?

A It's not immediately faster than the alternatives, which disappointed many early adopters. Nor is Wayland network-transparent, which means it doesn't include the ability to serve desktop sessions across a network in the way that early X11 did.

The complexity that comes with network transparency is a burden on the current system. That doesn't mean there won't be an alternative, such as a more VNC-like approach to sharing the image buffer to a remote address. It just means that any solution won't be as overbearing. In fact, the Wayland community think that a remote desktop solution using Wayland will be better than VNC on X11 anyway.

Q X11 is the system that Wayland is going to replace?

A Yes. X11 was designed for a different era of computing – the same 1980s of that old Commodore 64. And the key requirement for any 1980s-era enterprise computing installation was for remote graphical terminals. X11 was designed to work across a network so that low powered,

cheap terminals could connect to a centralised computing resource with lots of storage, CPU and RAM.

The apps ran on the central computing resource and sent instructions back on what to be displayed to each terminal. You can still do this with X11 today. In fact, you are doing this today: X11 uses the same client and server configuration even when everything is running on the same machine, making the separation between the client and server a little pointless. There are several other big chunks of X11 that have become redundant, such as its inclusion of some core fonts or big parts of the rendering API – features that are now part of toolkits like GTK and Qt. Then there are the four input subsystems. And that network transparency we were just talking about? It won't work when using modern modern systems with X11 anyway, because of the way they talk to the local graphics hardware.

Q So what's to stop Wayland from being just as bad?

A Apart from the simplicity, there's no legacy code to get in the way of creating a modern graphical subsystem. Just imagine what X11 might be like in another 10 years, and it's difficult to think of how it might adapt to tablets and smartphones.

It's also capable of using hardware specific backends. This won't be necessary for most installations, but there's a backend for the Raspberry Pi that has considerably improved its graphical prowess, when compared to X11, so perhaps performance might improve after all.

Q If it's so hopelessly crufty, how has X11 lasted so long?

A For one simple reason: it works. That's something that can't be said for a great many other technologies. It's stable, despite its complexity, and it's a well understood and a well integrated part of the system. Thanks to the development of many other modules that connect to X11, it's a modern and adaptable solution. But it's never going to get simpler nor better adapted to the kind of computing we do now. All those extensions and plugins, for example, lack any kind of version control, and that means there's no easy way of knowing which features you're going to get when your application supports a different version of a plugin to your X11 installation. Come the revolution, though, you'll still be able to run X11 tools and applications through a compatibility layer called XWayland.

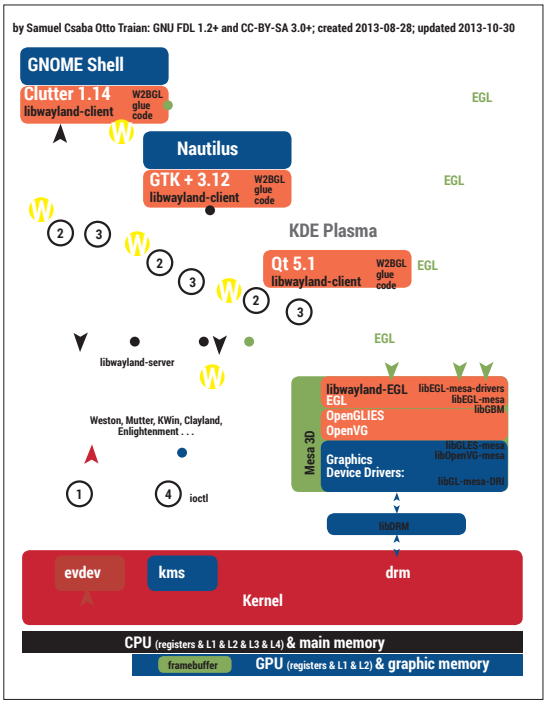
Q Does Wayland do away with the server and client model?

A No. But the client/server model used by Wayland makes more sense. The server is something called the Wayland Compositor, and desktops such as E17 or Gnome, going through their respective APIs, are considered the clients. This is why you always see Wayland described as a 'protocol', rather than a way of rendering graphics. It's the protocol that defines how the clients speak to the server. A server could be replaced with another server, as long as they understood the same protocol. Which is exactly what the people behind Wayland hope will happen. E17 and Gnome both have their own Wayland-compatible compositors.

Q Hang on a mo – what's a compositor?

A You might have first heard the term when desktop effects started to become popular. Compiz, for example, is perhaps the best-known compositing window manager. It adds effects such as wobbly windows, desktop shadows and transparency, and it does this by compositing the contents of the various windows under its control into a single image that can then be used as the desktop. That's why when you run Compiz, you have to replace whatever window manager you're currently using.

In Wayland, the compositor does the same job, only without the help of X11 to turn the final composited image into the desktop you see. It's the server process that pulls all the graphical components together to create what you'd expect to see on the screen. For Wayland, that would mean the server that composes the contents of the various client application windows before sending them on to the rendering stage. These elements already exist, and are not part of Wayland. They're used to get the output from the compositor to your screen.



Wayland is mostly a protocol because it defines how the various components in the stack talk to one another [Image Credit: CC-BY-SA 3.0: ScotXW, based on work by en:Kristian Høgsberg published at en:freedesktop.org: http://wayland.freedesktop.org.

Q If Compiz worked with X11, why is Wayland any better?

A It simplifies the process. X11 was the gateway between the app and the compositing. With Wayland, the applications talk to the compositor without having to go through X11. Wobbly windows with a compositor in X11 worked, but it was much harder if you wanted to tell the compositor you were working with hardware overlays to play back video. Wayland's direct line of communication is a much better way to accomplish the same tasks.

Q Is 2014 going to be the year of Wayland on the desktop?

A We think so, yes. Gnome 3.12 can now operate as a Wayland compositor, bringing native support to the Gnome desktop. So too can Enlightenment's compositor after a huge code dump in the middle of January. The reference compositor for Wayland, named Weston, saw plenty of updates in January's 1.4 release, and there's a new Qt 5.2 based desktop called Hawaii that uses Weston. Even if KDE's Wayland support is slow in making an appearance, you'll definitely be able to migrate to a Wayland-only desktop in the near future.

SYSADMIN

System administration technologies brought to you from the coalface of Linux.



Jonathan Roberts dropped out of an MA in Theology to work with Linux. A Fedora advocate and systems administrator, we hear his calming tones whenever we're stuck with something hard.

This is the first issue of *Linux Voice*, and I'm so pleased to be contributing to it along with Graham, Andrew, Mike and Ben. It feels new and exciting, with all four of them having poured so much enthusiasm into the project, it can only work out well.

When Graham asked me to write the first two pages of this new sysadmin section, I started looking around for ideas, and I noticed that in the same way *Linux Voice* represents a kind of transformation of a well loved thing, so Linux itself seems to be undergoing a similar transformation.

New technologies, like systemd (and its many associated subprojects, including journald and logind), btrfs, cgroups and kdbus are slowly replacing older technologies and approaches that many people have long assumed to be synonymous with Linux.

Many sysadmins have been ignoring these new technologies, in part because of all the controversies surrounding them, and in part because of inertia. But the times, they are a changing.

Something better change

With the announcement of Red Hat Enterprise Linux 7 Beta, with the emerging plans in Debian to replace its init system, and openSuse having seriously debated btrfs as its default filesystem, the time has come for sysadmins to start taking these technologies seriously, as the next round of major distro releases are certainly going to include some combination of them.

So, a new and exciting magazine and a new and exciting set of technologies – seems like the perfect chance to take a closer look. Look out for my coverage of some of these new technologies over the next few issues of Linux Voice.

systemd

Don't fear change

The init replacement for RHEL 7 and SUSE Enterprise Linux 12.

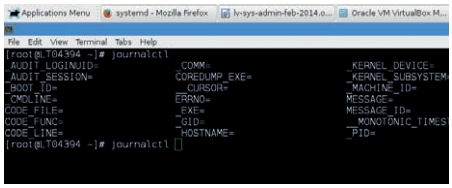
The arrival of a new Linux init system has been a long time coming. It was back in 2006 that Upstart was introduced to Ubuntu, and around the same time that Fedora and others also started experimenting with new init systems. The reasons then are much the same as the reasons now – sysvinit is old and doesn't do everything a modern distribution needs it to. More specifically:

- **sysvinit** can't take account of hot-pluggable hardware devices and filesystems, such as network mounts or USB sticks.
- **sysvinit** doesn't provide sufficient supervision of processes, allowing double forked processes to become orphaned.
- **sysvinit** can't parallelise boot services effectively, so it is slow.
- **sysvinit** startup scripts are difficult to write, difficult to debug and can't easily be shared between distributions – the Sendmail init script is over 1,000 lines long!

Systemd fixes these problems and introduces a number of new features that make the case for it even more compelling. Rather than explaining in great detail how systemd works or how it fixes these problems (there's plenty of information on that in **http://0pointer.de/blog/projects/systemd.html**), we're going to take a look at a few key features of systemd that might make sysadmins look forward to systemd, rather than dread having to learn a new tool.

Configuration file format

As mentioned above, in sysvinit systems, configuration of services was complex and error-prone. They were usually configured through a combination of arcane Bash scripts in **/etc/init.d** and some environmental settings in **/etc/sysconfig** or **/etc/defaults**. These init scripts often did awful amounts of work, such as echoing service status to the console and managing lock files, which were repeated in almost every init script.



Most of systemd's tools feature tab-completed sub-commands, which is indicative of the efort that's gone into making it a pleasure to use.

Systemd removes the need for much of the complexity in these init scripts by handling service status echoes and suchlike itself. This means it can switch complex procedural Bash code for a clear, declarative configuration file. For example, here's the configuration for the syslog service on my Fedora system:

```
[Unit]
Description=System Logging Service
[Service]
EnvironmentFile=-/etc/sysconfig/rsyslog
ExecStart=/sbin/rsyslogd -n $SYSLOGD_OPTIONS
Sockets=syslog.socket
StandardOutput=null
[Install]
WantedBy=multi-user.target
Alias=syslog.service
```

All of the configuration options available in these files are extremely well documented (systemd as a whole has some of the best docs around) – see **man systemd.unit** or **man systemd.service** for details.

What's more, if you had to modify a sysvinit file, you'd have to be careful when it came to package upgrades etc that your changes wouldn't get overwritten. With systemd, unit files get packaged into **/usr/lib/systemd/system**, but if you want to replace the default with your own, you can

put them in **/etc/systemd/system** and whatever is there will take precedence over the defaults.

You can even include other unit configuration files in yours, so you can easily extend the default configuration:

```
include /usr/lib/systemd/system/nfs-secure.service
#extra conf goes here
```

Resource controls

Why would you want to extend a service configuration like that? Well, systemd launches all processes inside their own cgroup (and all processes spawned from this end up in the same cgroup – this is also useful as it stops double forking processes from orphaning themselves), so you can take advantage of this to use cgroups to limit the resources that each process (and its child processes) can consume.

Systemd not only makes this possible by the way it spawns processes, but it also makes it easy by exposing many of the most common bits of functionality in configuration directives. For instance, you could limit the amount of CPU a process gets by dropping in a new unit configuration file to **/etc/systemd/system** and adding:

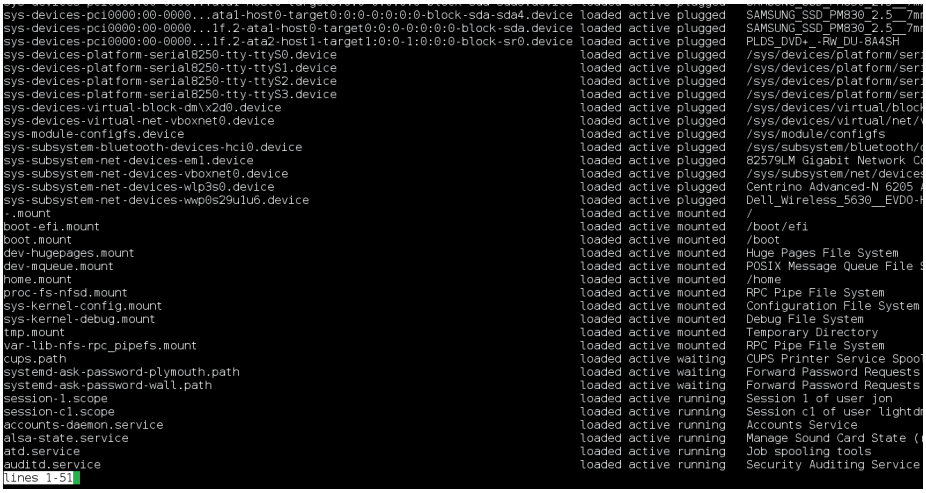
```
[Service]
CpuShares=200
```

By default, systemd gives all processes (well, cgroups), an equal share of the processor (1024). By setting **CpuShares** to 200, you're restricting this process to about 20% of CPU time. What's more, this isn't applied just to the parent process but to all child processes. So if you have Apache running with many hundreds of spawned CGI processes, this would restrict all of those processes to about 20% of CPU time.

With the configuration file in place, you'd just need to tell systemd to reload it, with **systemctl daemon-reload**, and then restart the service, with **systemctl restart httpd.service**, for example.

You can also set memory limits (MemoryLimit) and IO limits (BlockIOWeight). See **man systemd.resource-control** for further details. There are also any number of security settings that can be put in the configuration files like this. For example, you can restrict a service from accessing a particular device, make individual directory trees inaccessible or read-only, create a private **/tmp** directory for a service or even stop a service, and all its child processes, from accessing the network. In the example below, we've configured a service to have a private **/tmp** directory. See how simple it is:

```
[Service]
PrivateTmp=yes
```



sysctemctl enables a user to easily inspect what units (services, in systemd speak) are loaded on their system and what their current status is.

displayed, just as if you were looking at the contents of **/var/log/messages** or similar.

This default view gives you some simple improvements over the traditional techniques, however. Error and higher priority messages are in red, notice and warning are bold, timestamps are in your local timezone. These are fairly cosmetic improvements. What sets journald apart is that the logs are kept on disk in a binary format, which means that the journal entries can be indexed on all fields, making them quick to search and easy to filter. For example:

```
journalctl PRIORITY=7 –since=yesterday
```

Will show all messages of debug priority received by the journal since yesterday. If you tried to do this with standard syslog messages or the like, you'd have to concoct your own grep or awk command, or hook it in to a system like Logstash or Splunk.

There are loads of fields on which you can filter that come direct from the messages themselves, as well as a lot of metadata that the journal inputs in to each log message itself, including SELinux context, hostname, transport etc. To see the full details, you can read **man systemd.journal-fields**. Journalctl even features tab completion of possible field names, so you can get a quick look too by typing

```
journalctl <tab><tab>.
```

There are many other great features in systemd that, if you take the time to look around, will make your life as a sysadmin better. We hope this article has at least given you the motivation to take a closer look.

> **"Another aspect of systemd is that it collects all output from processes that it starts."**

Journal

Another aspect of systemd is that it collects all output from processes it starts – whether that's through **syslog()** calls, messages emitted to STDOUT or STDERR, initial RAM disk or kernel messages. It does this through one of its components, journald. To see the contents of the logs, you can just type **journalctl** as root and you'll get the results



The other contender for the next generation of Linux init systems, Upstart, is the Ubuntu's choice and involved in a hotly contested debate in the Debian project over what they should choose.

CLOUDADMIN

Nick Veitch opens your eyes to the technology behind the cloud server revolution

Openstack

The cloud at last

All DevOps is headed to the clouds, but that doesn't mean abandoning open source platforms.

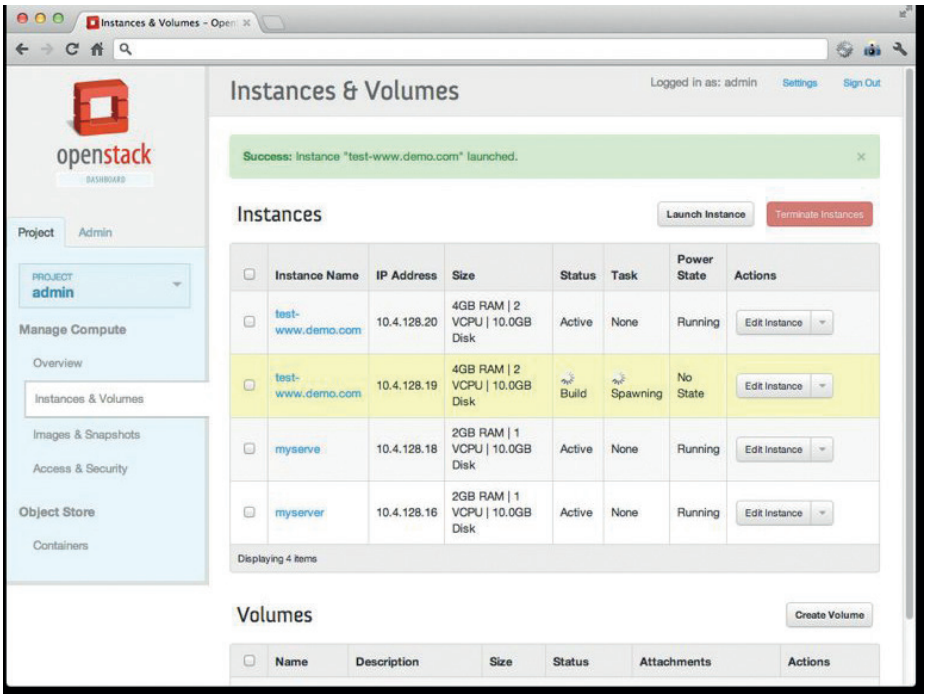
People used to eye the sky nervously and then fix you with a slightly menacing stare when you mentioned you had anything to do with clouds. Fortunately it is now a well-known word bandied about by IT types. Last year, Forbes reported that over half of US businesses used cloud services in some form or another (**www.forbes.com/sites/reuvencohen/2013/04/16/the-cloud-hits-the-mainstream-more-than-half-of-u-s-businesses-now-use-cloud-computing**), and of course, indirectly many consumers do too, whether it is for simple file storage, backup of mobile devices or cloud-based music streaming. In fact it is pretty hard to escape clouds, even if you want to.

So what is cloud computing?

The basics behind cloud computing are that it is an elastic resource, a commodity, where you can consume, on demand, resources such as computing power and storage. As such it has been highly disruptive – it isn't by accident that the pioneers in much of this computing-as-a-commodity revolution are an online bookstore rather than one of the major IT players. Though of course, Amazon now is a major IT player.

The good news for open source in all this is that the up and coming platform of choice with which to build OpenStack. The project began life as a collaboration between NASA and the hosting company Rackspace. The idea was simply to build a suite of open source tools that could provide the infrastructure to deliver cloud services, whether that be a private, internal, cloud (such as the original NASA project on which OpenStack is based) or a public-facing cloud, like the one developed by HP (**www.hpcloud.com**).

> **"With our OpenStack cloud, we've been able to offer physicists the resources they need to analyse and model data from the Large Hadron Collider."**



The OpenStack dashboard can be used to manage every aspect of your cloud deployments, and it's also possible to control it through an API, for great automation wins.

Why open matters

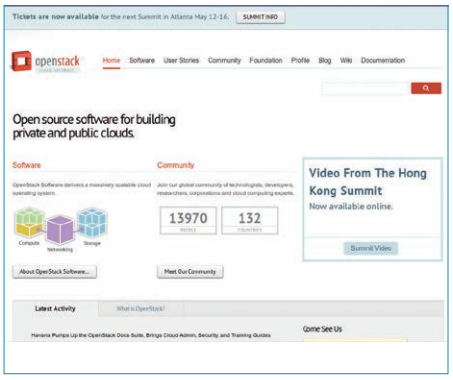
Openness in the cloud matters just as much as openness in an operating system. In fact, a cloud platform performs very much the same function as an OS – it enables the user to deploy software to perform some task. On a desktop, that might be image editing in Gimp; on a cloud it might be hosting a global payment platform or similar. With an open solution, it is at least theoretically possible to move your workload from one provider to another.

That may not seem to matter much (and indeed, it is currently difficult to achieve) but it will matter more as startups may need to shift workloads from local hardware to a more resilient cloud provider, or larger companies invest in hardware and pull services out of public clouds. It also provides a level playing field for innovation – many companies now contributing to OpenStack are developing a powerful and rich ecosystem of possibility. These companies include huge players in the IT field. The list is quite long and includes names like

Cloud anatomy

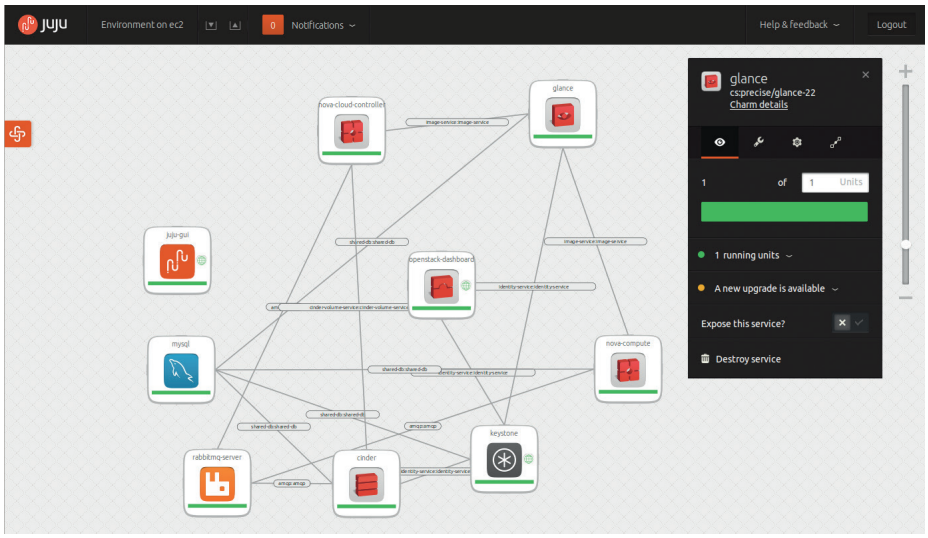
OpenStack is a suite of software tools designed for specific purposes. Each workload, and perhaps an entire cloud, may not need all of the components. New components are also being developed as the need for them arises. Here are the current ones (and their OpenStack names!)

- **Compute (Nova)** One of the primary requirements of any work you may want to do in the cloud, the Nova component allocates compute resources (ie processing power to run software). Nova is designed to be very efficient at scaling horizontally (across many nodes, rather than adding more resources to a single node).
- **Object Storage (Swift)** Object storage is storage that can store and retrieve blobs of data. They might be big blobs- a huge video file for example, or little blobs, like the *Linux Voice* payroll. The advantage of this sort of storage is that it is easily mirrored and replicated, and may be split easily across many actual nodes.
- **Block Storage (Cinder)** This type of storage simulates a block-level device, such as you would find attached to your **/mnt** directory in Linux. As such it is transparent to applications accessing it and can be much faster than object storage, but it also requires slightly more overhead in resources. Like ordinary block devices, it needs a filesytem.
- **Image server (Glance)** In dynamic clouds where nodes might be scaled out at any time, it is useful to have a store for images of particular devices (eg. a backup of a virtual server or a filesystem). This is a sort of specialised version of Swift (indeed, Glance often uses Swift as its storage medium), but with extra functionality, such as an API for querying what images are stored.
- **Networking (Neutron)** Software-defined networking was first cooked up in 2008, and was joined later by Network Functions Virtualisation. Simply put, in an age where complex workloads are being defined in software, it makes sense to create the network topologies there too (Virtual LANs), and indeed virtualise other functionality (intrusion detection, DNS etc).
- **Identity (Keystone)** Managing access to cloud services is a key concern. Keystone is a directory mapping users (and software) access to the services deployed. Authentication can be configured to accept a variety of methods – passwords, tokens, etc.
- **Dashboard (Horizon)** The dashboard or control panel is just an easy to use interface for tweaking the cloud (or your part of it) as well as controlling the automation of some services. It is of course possible to use software tools to access all of this functionality through the API.
- **Metering (Ceilometer)** Openstack describes this as a telemetry service, but in simpler terms it is basically the equivalent of an electricity meter, collecting data on what was used when and by whom. Although designed primarily for billing users, it provides auditing information for plenty of other purposes.



There is a very excellent website for OpenStack at www.openstack.org, and a very useful wiki at wiki.openstack.org.

Rackspace, HP, IBM, AT&T, Canonical, Dell, Cisco, SUSE, Intel, VMWare, RedHat, NEC, Hitachi and more.



The Canonical's Juju GUI view of a deployed OpenStack instance. Other configurations are possible! Visit https://jujucharms.com and search for OpenStack bundles.

Loads of work

In many ways it is not the platform that is so interesting as the workloads that are deployed on it. There are inherent advantages to deploying even traditional server solutions on a cloud. If you run a simple LAMP stack on the cloud, it can automatically take advantages of features that the cloud provides for it – the underlying platform can make it more resilient, and vertically scale resources (storage, processing power) if required. There are more benefits to realise for software that has some mechanism for horizontally scaling – distributed, highly-available applications that run on multiple nodes. In future workloads will be written specifically for cloud use and take better advantage of the provided APIs to dynamically control aspects of their deployment.

Not everyone is a fan of OpenStack. Analysts at Gartner have continually downplayed the success of the platform, suggesting that there is a lack of differentiation and confusion about the services offered by various participants, and that the business model isn't clear. Uncannily, that is almost exactly what they used to say about Linux...

It is true that OpenStack is maybe not as mature as Amazon's Web Services, but the project is just over three years old, and there are a ridiculous number of companies committed to driving it forward. If you want to play with the clouds of the future, OpenStack is a great place to start.

FOSSpicks

Sparkling gems and new releases from the world of Free and Open Source Software



Mike Saunders has spent a decade mining the internet for open source treasures. Here's the result of his latest haul…

Connection problem solver

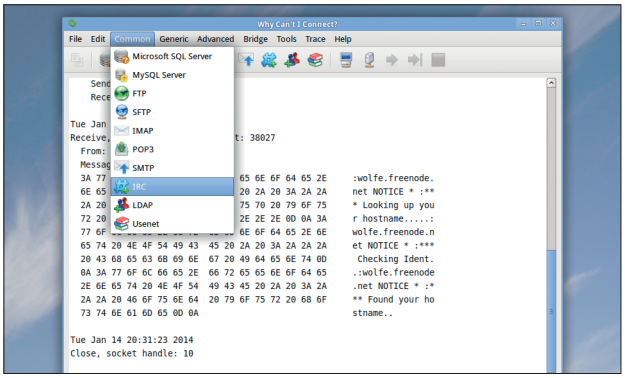
WHY CAN'T I CONNECT 1.6.2

http://wciconnect.sourceforge.net

Gimp, Firefox, Evolution, Scribus – all great free software projects, but their names say very little about what they actually do. That's not a criticism we can level at Why Can't I Connect (WCIC), however, because it does exactly what it says on the tin. WCIC attempts to help you pinpoint problems when you're connecting to servers across the internet, providing more information than you might otherwise receive from your usual applications.

Say for instance that you're trying to set up a mail program, and you can't connect to an IMAP server. Your mail client might just throw up a generic "Connection failed" error message, which is about as much use as a chocolate teapot. Where exactly is the connection failing? Is the problem with the IMAP server, or something else in between on the net? If you're massively knowledgeable about the particular protocol you're using, you might be able to Telnet into the remote machine and work it out from there, but not everyone can do this.

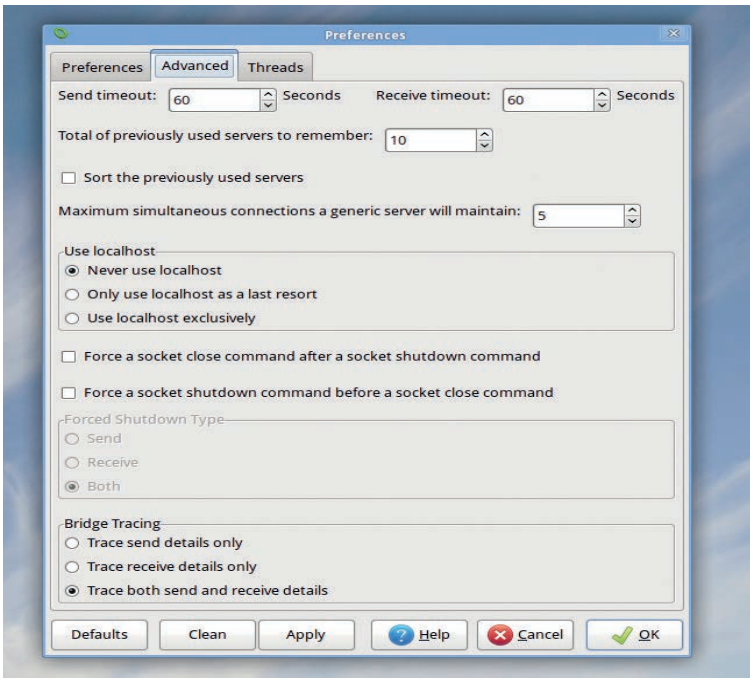
WCIC makes things much easier. You tell it to initiate a connection with a remote machine, and it steps through the process, spitting out lots of feedback and detailed descriptions of error messages. It supports a bunch of protocols out of the box: MS SQL, MySQL, (S)FTP, SMTP, IMAP, POP3, IRC, NNTP and LDAP. Just supply a hostname/IP address and a port number, and WCIC will sort out the rest.



Never struggle with lame error messages again: WCIC tells you what's really happening with your connections.

If you want to investigate connection problems with a different protocol, you can go into the Advanced menu and look through the connection process step-by-step, sending textual data from a file to see how the remote server responds. It's even possible to set up WCIC as a bridge between a client and a server, so you can look at the data that the machines are exchanging in real time.

WCIC is one of those helpful tools that we'd love to see included by default in mainstream desktop distributions - so make it happen, distro devs!



Set timeout limits and fine-tune how WCIC interacts with remote servers via the Preferences pane.

Free-form data organiser

TREESHEETS 26-10-2013

http://strlen.com/treesheets

Take a spreadsheet, mix in some mind-mapping elements, add a few to-do list and text-editor features, and what do you get? The result might sound like a hideous mish-mash of programs, but it's actually a rather awesome fusion that works really well. TreeSheets' author describes it as "suitable for any kind of data organisation", such as calendars, to-do lists, project management charts and brainstorming diagrams.

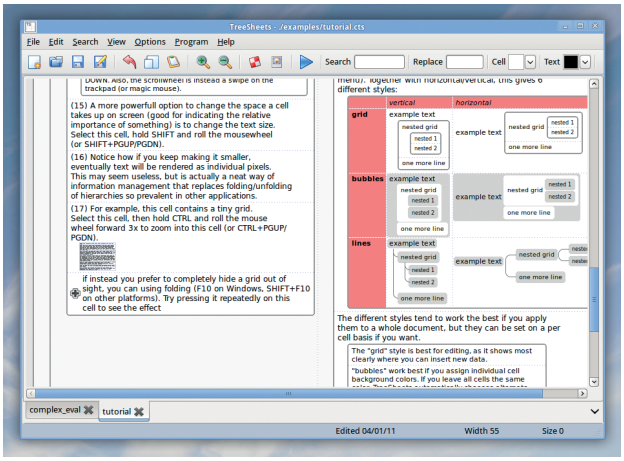
The key to all this flexibility is hierarchy: items can be embedded inside other items.To get it running you just need to grab the **treesheets_linux.tar.gz** file and extract it – this produces a directory called **TS**. In here, run **./treesheets**, a pre-compiled binary. This worked without any fuss on our Xubuntu 13.10 test machine, and providing you have GTK 2 and its dependencies installed, you shouldn't have any problems. (If the binary doesn't work, you can still compile from source.)

Excellently, TreeSheets is supplied with an in-depth tutorial that doesn't just show you how to do things, but encourages you to experiment by hand. It's shown automatically on the first launch; to bring it up at another time, hit F1. Essentially, TreeSheets works as a souped-up spreadsheet, letting you enter data into cells and move them around. Click on the edge of a cell and its border turns into a dotted line: start typing and new data is added into a new cell. Hit the Insert key (or go to Edit > Insert New Grid) to add data to a new table – like a spreadsheet inside a spreadsheet.

Lookin' good

You can format these grids in various ways via Edit > Layout Render Style. With these features alone you can come up with novel ways to present data such as contacts lists, to-do lists, project plans and even algorithms – see the **examples/** folder for inspiration.

But TreeSheets can do a lot more. It's possible to filter content based on edits, eg showing only the cells that have been edited in this session. The early phases of a simple programming system have been included, where cells can be marked as operations (eg addition or multiplication) and then generate results from neighbouring cells. Once you have everything in place, click the Run button on the toolbar to calculate results. The **complex_eval.cts** example sheet demonstrates these features well.

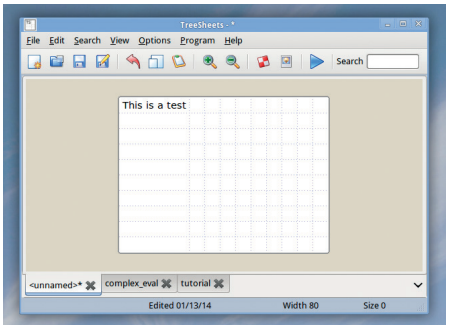


The built-in tutorial is one of the best we've ever seen, getting you hands-on with the program right from the first step.

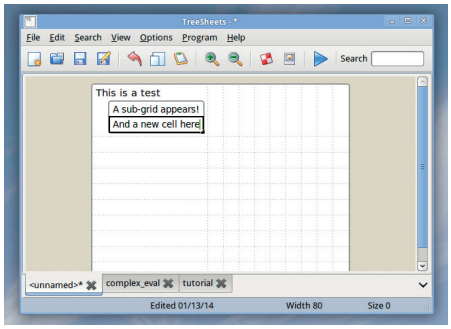
> **"The key to TreeSheets' flexibility is hierarchy: items can be embedded inside other items"**

Ultimately, TreeSheets is only limited by your imagination. It might seem like an app without a specific purpose at first, but once you've browsed the examples you'll see how the program's features could be applied to your own data. We've already started using it as a to-do list manager on steroids, and as the programming features get more fleshed-out, its range of uses will be nigh-on endless.

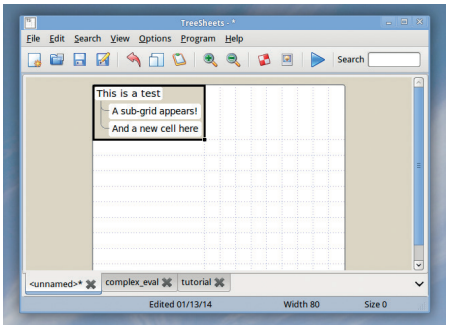
How it works: Adding data



1 Create a grid
Click on File > New to create a blank 10x10 cell grid. Start typing, and the first cell will expand to fit the data that you enter.



2 Add a sub-grid
Click Edit > Insert New Grid to create a grid inside the current one. Type some text, click on the lower edge, and type again to add a new cell.



3 Format it
Click on blank space in the outer grid, then Edit > Layout Render Style > Vertical Layout With Line Style to display the items in a tree-like format.

Graphical disassembler

EMILPRO 3.0

www.emilpro.com

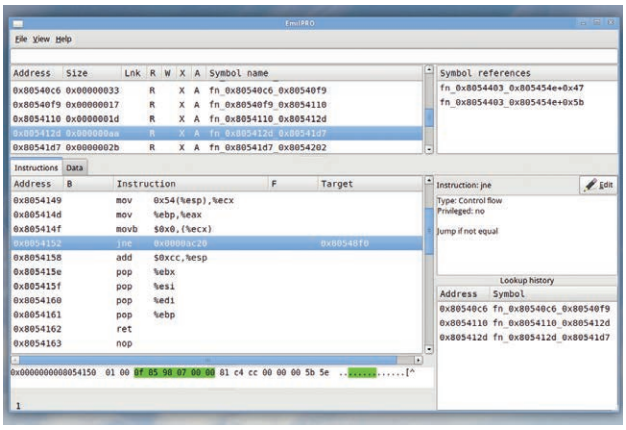
Some under-the-hood things are worth understanding, even if they're not especially useful on a day-to-day basis. One of these, if you're a programmer, is assembly language. Sure, very few people write substantial code in assembly nowadays (apart from embedded device and driver developers), but it's worth understanding what goes on after you compile your high-level code.

EmilPRO is a GUI disassembler: it takes binary machine code executables, identifies the chunks that are code (instead of data), and shows the human-readable CPU instructions contained therein.

The README file explains the dependencies required and how to compile the program. Once you have EmilPRO up and running, go to File > Open and locate a binary executable on your filesystem. On a Linux box this executable will be in the ELF format, but EmilPRO can also read Windows and Mac OS X executables using its own version of Binutils, which it builds during the compilation process.

The interface

The top-left panel shows a list of sections and symbols in the executable, while the panel to the right refers to the currently selected symbol. Underneath, on the left, you have the disassembled code (or a hex dump if it's a data section), and the right-hand side shows some information about the currently selected instruction. We're not sure how useful this is – after all, if you don't know what the instructions mean, you're probably not interested in looking at assembly language – but from inside the app you can submit more relevant descriptions to EmilPRO's website.



Looking inside /bin/ls: the code here shows a subroutine, which hands control back to the calling code with the "ret" (return) instruction.

EmilPRO works well enough, although the interface could do with some polish. The program's author admits that he's not an expert with GTK, so if you are, dear LV reader, hop over to **https://github.com/SimonKagstrom/emilpro/wiki/TODO** to lend a helping hand.

Super-simple cron job creator

EVERY 0.1.0

https://github.com/iarna/App-Every

One of the best things about Linux (and most Unix-like systems in general) is the widespread use of plain text configuration files. Look in **/etc**, for instance, and you'll see that almost everything is readable and editable in a standard text editor. Contrast this to Windows, with its dreaded registry... Ugh. Let's not go there.

It's not a total utopia in Linux though: most configuration files are text-based, but the syntax and formatting in them can vary wildly. If you're a seasoned Linux admin then you can probably write cron entries in your sleep – but if you always end up having to consult the manual page, Every is for you. It's a small Perl script that lets you enter human-readable descriptions of tasks that should be executed at specified intervals, and it updates your crontab accordingly.

Get the latest release like so:

```
wget 'https://raw.github.com/iarna/App-Every/master/packed/every'
```

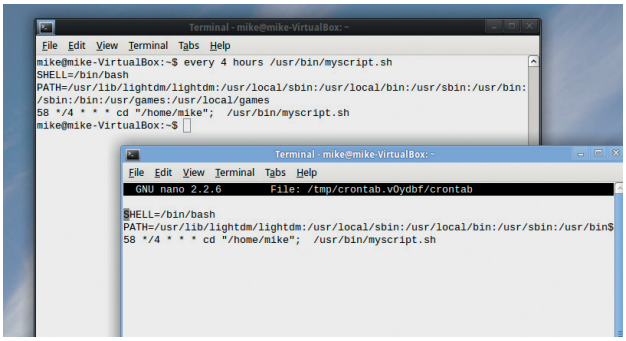
Then make it executable (**chmod +x every**) and copy it into **/usr/bin** or somewhere else in your **$PATH**. Using it is really simple: let's say that you want to run **/usr/bin/myscript.sh** every hour. All you need to enter is this:

```
every hour /usr/bin/myscript.sh
```

And that's it – you don't get much more human readable than that, right? You can replace "hour" with "minute", "day", "week" or "month", and even specify a day of the week (eg "every thursday"). Additionally, you can specify units like so:

```
every 23 minutes /usr/bin/myscript.sh
```

When you run Every, you'll see that it spits out a crontab entry, and it also automatically updates the crontab for your user account (edit it manually with **crontab -e**). Every sets up the $PATH and your filesystem location exactly, so you can guarantee that the command will run exactly as intended.



Every shows you its generated entry before adding it to the crontab – use the --dry-run option to preview the entry.

"Most config files are text-based, but their syntax can vary wildly"

Log file monitor

BEOBACHTER 1.7.8

http://sourceforge.net/projects/beobachter

Keeping tabs on log files is just one of the many jobs a hard-working sysadmin has to do. The trusty **tail** command line tool (with the **-f** flag) does a decent job here, but it's plain, and doesn't help you to spot critical information when text is whizzing by. A more elegant solution is to use a log file monitor, and Beobachter (from the German "*beobachten*, to observe) has a few aces up its sleeve.

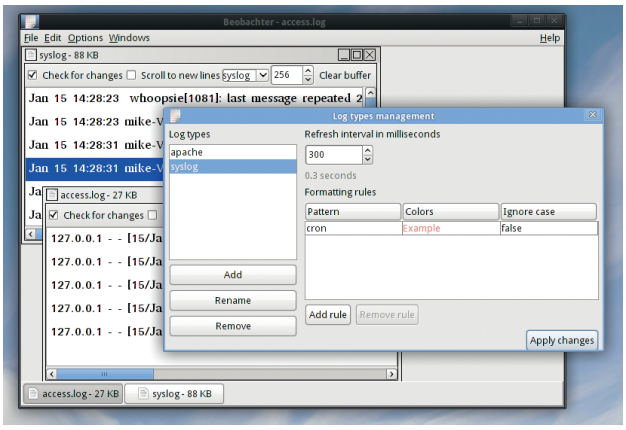
To run Beobachter, grab the **.jar** file and enter:

```
java -jar Beobachter-1.7.8-jar-with-dependencies.jar
```

Click File > Open and select a log file (eg **/var/log/syslog**). Beobachter will then show updates to the log file in real time, so as soon as anything is added to the log, you'll see it in the window.

Beobachter can view multiple log files simultaneously; you can save your session (so that all your log files are re-opened when you next start the app) via the File menu.

Where Beobachter really comes to life is with the formatting rules. Under Options > Manage Log Types you can create custom rules to apply to specific log files (or groups of log files), so that lines are highlighted when they contain specified words or regular expressions. For instance, you might be watching a log that's updated often, so you don't have time to read every line, but you want to be alerted when the word "error" crops up.



Its default interface is very ugly (it's written in Java), but you can tell it to adopt some elements of your GTK theme in Preferences.

> **"Where Beobachter really comes to life is with the formatting rules"**

With the formatting rules feature, you could set lines that contain the word "error" to have a red background, so they're easier to spot. After some tweaking, you can have a full-screen Beobachter session set up with multiple log files being viewed and errors leaping out of the screen. Not only does this make your life as an admin easier, it looks pretty l33t too.

Browse inside zip files

FUSE-ZIP 0.4.0

http://code.google.com/p/fuse-zip

Most modern file managers let you browse inside compressed archives. Gnome and KDE have services to give seamless access to compressed archives, for instance, but standalone non-Gnome/KDE programs can't use them without pulling in loads of dependencies.

fuse-zip fixes this problem by letting you manually mount **.zip** archives as normal directories, without the need for any major dependencies. The only thing it requires is FUSE (Filesystem in Userspace) which is included by default in many major distributions. To build fuse-zip, you'll need at least version 2.7 of **libfuse-dev** and version 0.11 of **libzip-dev**. On our Xubuntu 13.10 test box, entering **make release** to build it resulted in an error that UINT16_MAX was not defined; we fixed it by adding this to the top of **lib/fileNode.cpp**:

```
#define UINT16_MAX 65535
```

Once you have it built, run **make install** as root and you're ready to go. Mounting a zip file in a directory is as easy as pie:

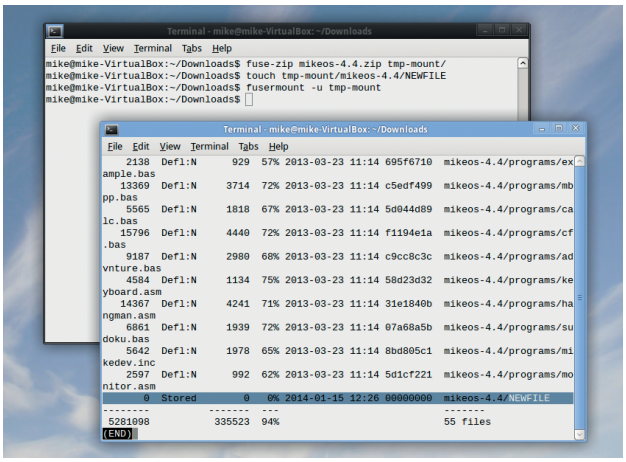
```
mkdir mytmp
fuse-zip somefile.zip mytmp
```

Now you'll be able to seamlessly browse the contents of **somefile.zip** via the **mytmp** directory. Any edits that you make to files will be stored back in the **.zip** file when you unmount the directory, which you can do like this:

```
fusermount -u mytmp
```

fuse-zip's developer claims that it's faster than the Gnome and KDE equivalents, with plenty of stats to back it up. You can even use it to create new zip files, eg **fuse-zip newfile.zip some-directory**. When you unmount **some-directory**, **newfile.zip** will be created.

It all runs like clockwork, so if you've been avoiding the heavyweight desktop environments but you still like this feature of their file managers, give fuse-zip a go.



In the background terminal we mount mikeos-4.4.zip, add NEWFILE to it, and unmount it. And the foreground terminal shows it has been added to the .zip. Awesome.

Crazily tiny PC emulator

CABLE3

http://ioccc.org/2013/cable3

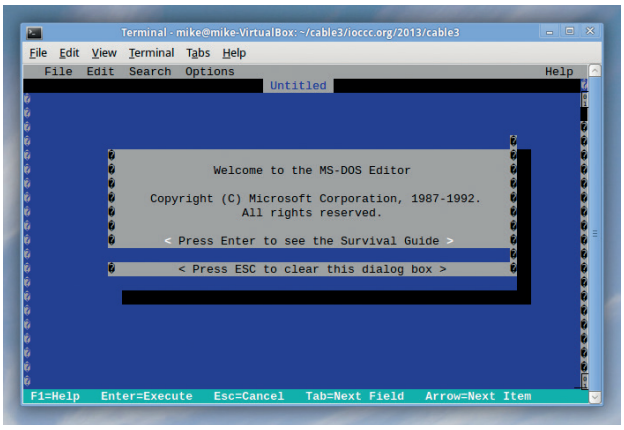
Look at this snippet of source code and see if you can guess what language it is

```
*i+=262*o*z(F((*E&15)>9|42[E])),*E&=15))
i(SP,(w(7),R&&--
1[i]&&o?R++,Q&&Q++,M--:0))DX()
{$,O*=27840;O--;
```

If you're feeling particularly snarky, you might say "It's Perl, with some extra formatting to make it more readable than a typical Perl program". But no, it's actually C - and completely valid C. Each year, the International Obfuscated C Code Contest (**www.ioccc.org**) asks programmers to submit the most weird, messed-up and preprocessor-abusing code possible, and some of the results are truly astounding.

The above snippet is from Cable3, a fully fledged PC emulator that's generated from a single 4043-byte C source code file. It's the obfuscation that keeps the code so small – in human-readable C it'd be a lot bigger – but regardless of how the code gets mangled, it's an impressive project nonetheless. All you need to build it is the SDL development libraries, so once you have them installed, just enter **make**. You can then run the emulator by supplying a BIOS image and a floppy disk image at the command line; see the **runme** script for an example.

Cable3 emulates a mid-1980s-era PC, complete with an 80186 CPU, 1MB of RAM, a Hercules graphics card, a floppy drive and a hard drive controller. So it's not particularly powerful compared to the likes of QEMU and VirtualBox, but it's capable of running lots of programs such as early versions of Lotus 1-2-3 and Flight Simulator. It's even possible to coerce Windows 3.0 into running – don't expect great performance though.



It's a bit rough around the edges, but squeezing an entire PC emulator into 4k of code is an astonishing achievement.

> **"Cable3 emulates a mid-1980s-era PC, complete with an 80186 CPU"**

MS Paint-esque graphics editor

PAINT.JAVA 0.9C

https://github.com/HeroesGrave/Paint.JAVA

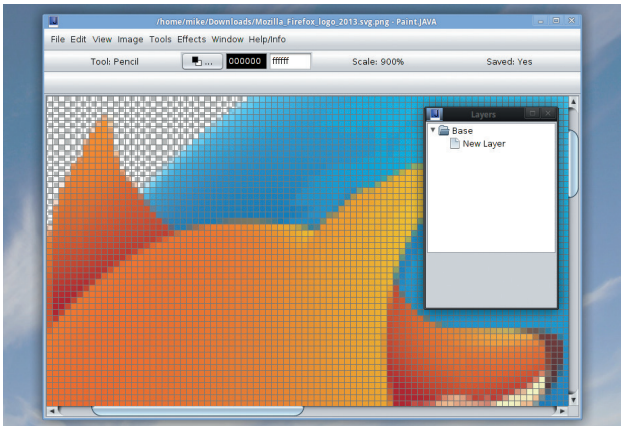
Microsoft Paint is a pretty rubbish image editor, but if you've ever spent time in Windows, you've probably used it in an emergency. It's basic, it's uncomplicated and it has hardly any features – but it's always there. An enhanced version called Paint.NET has been in development over the past few years, with a similar interface but much more functionality included, and now we have a Java-based clone of it.

Paint.JAVA was borne of frustration with existing pixel-editing tools. "Pinta is horrible, and Gimp is over-complicated and sucks at pixel art", says the developer. That's fair enough: Gimp is designed to be like Photoshop and focus on filters and effects for photos, rather than on plotting individual dots in images. So if you're designing sprites for video games or icons for programs and you haven't found any useful tools just yet, this could be a big help.

Although Paint.JAVA is still in beta, it's already quite usable. To run it, grab the **.jar** file, make sure you have a JRE (Java Runtime Environment) installed, and enter:

```
java -jar Paint.JAVA.jar
```

Don't expect much from the interface right now – it's very bare, and lacking the icon-laden toolbars you would expect to see in an image editor. Fortunately, there's still plenty of functionality tucked away in the menus, so click around and you'll find pencil and brush tools, shape creation tools, and an eraser. A handful of basic effect filters are included, and you can enable a grid mode, which helps when you're trying to keep various elements of a picture aligned.



Enabling a temporary grid helps enormously when you're plotting individual pixels and aligning things.

Paint.JAVA isn't the prettiest pixel editor out there, but it has all the basics covered and thanks to its Java roots it runs pretty much everywhere. With a few refinements and decent selection of plugins, it'll be a great little app one day.

FOSSPICKS BRAIN RELAXERS

Mascot-happy racing game

SUPERTUXKART 0.8.1

http://supertuxkart.sourceforge.net

SuperTuxKart has been doing the rounds for many years, so if you haven't played it for a while, you might remember it as a rough-edged Mario Kart clone with bits missing and gameplay flaws. That's certainly how we remembered it, until we tried the latest release. SuperTuxKart has come on in leaps and bounds over the last couple of years, thanks in part to its acceptance in Google's Summer of Code scheme – but also because it has a story mode now.

And this story mode is really well done: the GNU mascot (a gnu, unsurprisingly) gets kidnapped, and your job is to free him by winning races. The level of graphical and audio polish now approaches commercial games, especially during the intro, and the presentation is pretty slick too.

Still game

With 20 courses and 16 playable characters, there's plenty of variety in the game, and many of the courses unlock as you progress through the game. The controls are just about perfect now: the karts react well and don't slip and slide as they did in early releases. If you find the going tough, though, you can try a tutorial mode which takes you through the controls step-by-step.

It's often said that part-time, hobbyist Free Software developers can't produce games that rival commercial counterparts in terms of polish, but SuperTuxKart shows otherwise. It's fun, it's pretty, it's silly, and racing as Tux against the FreeBSD daemon represents what Free Software is all about: different projects often battling each other, but all with the same target in the end.



Yipes! The GNU mascot is being beamed up into a sinister spacecraft. Get your racing gloves on and free him…

Object-hunting fest

E.T. GAME 1.03

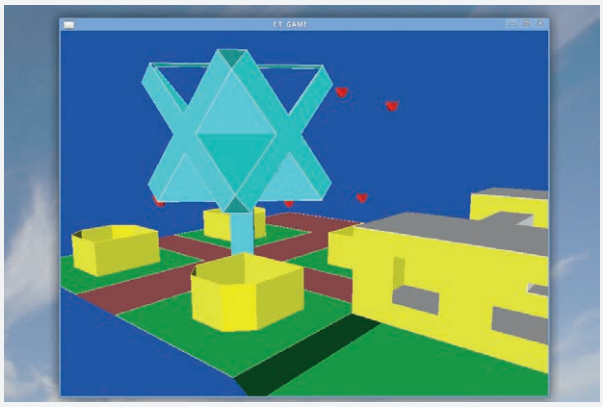
https://github.com/mki1967/et-game

Old games on Linux never die – they just need to be recompiled for the latest distros. Providing that the games are open source, of course. E.T. GAME hasn't changed much since its original release in 2003, but today the code is enjoying retirement in a GitHub repo, receiving the occasional update to make sure that it works with the latest versions of everything.

As you can see from the screenshot, E.T. GAME is totally retro in appearance – and this visual simplicity also means that it has very few dependencies. Providing that you have the development headers for X and GLX installed, you'll be able to get it built with a single **make** command. If the build is successful, enter **sh runme** to start playing.

Your goal is to collect 10 randomly placed red crystals that are scattered around the playing area. This is trickier than it looks, with some crystals tucked away inside rooms and crannies, and once you have them all you need to find a hidden exit. Nine levels are included – and there's even an editor to make your own.

"Old games on Linux never die – they just need to be recompiled"



Use the arrow keys to look around, and F and B to move forward and back. Hold Shift + left or right to sidestep instead of turn.

E.T. GAME won't last you long, but if you've ever fancied doing some 3D programming, the source code is worth exploring for ideas. It demonstrates how to get a simple 3D engine up and running without pulling in shedloads of dependencies, so you can easily modify it to turn it into a first-person shooter or maze exploration romp. LV



YOUR AD HERE



LV

Email andrew@linuxvoice.com to advertise here



LISTEN TO THE PODCAST

LINUX VOICE

WWW.LINUXVOICE.COM





LINUX VOICE

TUTORIALS

Dip your toe into a pool full of Linux knowledge with eight tutorials lovingly crafted to expand your Linux consciousness.



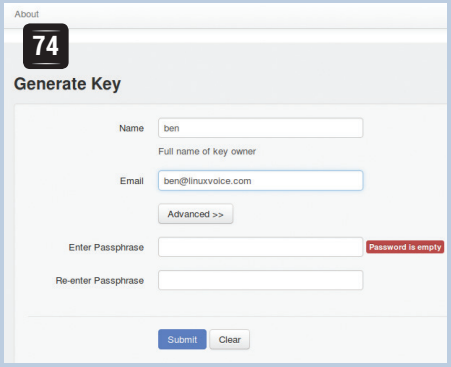
Ben Everard
has been busy brewing 'special' open source cider for the Linux Voice launch party.

There seems to be a link between Linux users and good beer. Of course, not all Linux users drink, and not all drinkers use Linux, but there's definitely a higher than average percentage of Linux users who like a really good pint. Quite naturally, geeks who like to tinker with their operating system also like to tinker with their beer. Graham's been going high-tech, and you can find out all about how to follow in his footsteps using the BrewPi kit and a Raspberry Pi to monitor the temperature of the brew on page 76.

I've been keeping things a little more low tech, but also experimenting to find the best brew. As I'm based in Gloucester, It seems appropriate that I've been focusing on cider. It tastes delicious, but can leave you with a sore head in the morning. This coming month I hope to try out some different malt combinations as I try to design my ultimate beer. I'll let you know how it goes – it should help me judge the results of the Bash regular expressions competition on page 102!

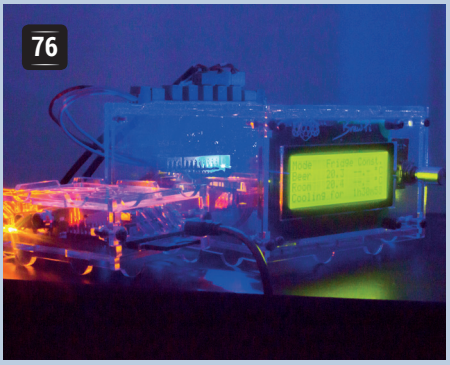
ben@linuxvoice.com

In this issue...



PGP

Keep your emails safe from prying eyes using state-of-the art encryption techniques – with **Ben Everard**.



BrewPi

Like beer? Like the Raspberry Pi? **Graham Morrison** uses the latter to keep an eye on the former as it ferments.



OwnCloud

Take control of your email, file storage and calendars – with **Mike Saunders**.



Ada Lovelace

Juliet Kemp tips her hat to this Victorian genius and progenitor of programming.



Arch

Take your first steps with with power users' distro – with **Graham Morrison**.

PROGRAMMING

Dynamic web pages

96 Perhaps the greatest thing about the web is how easy it is to quickly create a page to share information. However, keeping the data in your pages up to date can be a real chore, unless you have the skills to do it automatically. **Ben Everard** shows you four painless ways to keep information current.

Euclid's algorithm

100 We like to think of data encryption as being a modern invention, and it is, but the mathematical foundations on which it's built go back for centuries. **Graham Morrison** introduces an Algorithm from 3rd century Alexandria that can still be used to attack modern encryption methods.

Bash

102 The Linux command line is useful for many things: managing files, automating tasks, solving word puzzles. OK, that last one may not be the most common use of the GNU toolset, but it is a useful way of learning to work with regular expressions. **Ben Everard** sets some challenges to test your skills.

LINUX VOICE
TUTORIAL

KEEP MESSAGES SECURE WITH PGP

BEN EVERARD

The Feds (and GCHQ, and the NSA) are snooping on our communications, but we can fight back with encryption

Normal email is one of the least secure forms of communication available – less secure even than post cards. These mails can typically be read by anyone on the same network as you, anyone at the ISP, anyone at your mail provider, anyone at the recipient's ISP and anyone on the same network as the recipient, as well as anyone with access to the various networks between the two ISPs.

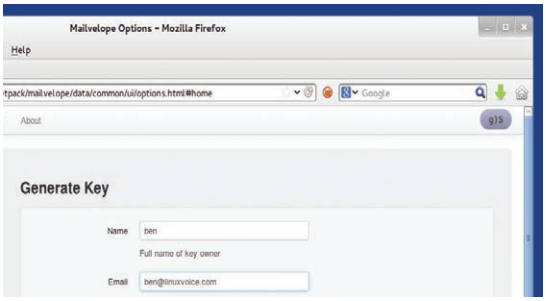
If you use SSL or TLS to connect to your inbox, then it improves things a little, but it's still vulnerable as soon as it leaves your mail provider.

PGP (Pretty Good Privacy) is a program designed to remove these weaknesses. It uses the normal email system, but adds a layer of encryption to protect them in transit. These days, PGP is usually used to refer to the OpenPGP format for these encrypted messages, rather than the PGP program specifically.

The OpenPGP format uses two different types of encryption: symmetric key and public key. In symmetric key encryption the same key (basically just a binary string that's used as a password) is used to encrypt and decrypt the message. In public key encryption, two different keys are used (one to encrypt and one to decrypt). The phrase 'private key' can refer to either the key in symmetric encryption, or the secret key in public key encryption. To avoid this ambiguity, we won't use the phrase in this article, but you may come across it in software.

When encrypting a message with an OpenPGP-compatible program, the software generates a random symmetric key and encrypts the text. This ciphertext forms the bulk of the message.

The problem is that the recipient of the message has to know the key, but it can't be included in the message otherwise anyone who intercepts the message will be able to read it. This is where public key encryption comes into play. Everyone who uses PGP first creates a public/secret key pair. The public key is made public while the secret key is known only to the user. However, anything encrypted with the public key can be decrypted only with the secret key and visa versa.



The colour and message in the top-right corner are a random security code so you can distinguish real Mailvelope messages from spoofs.

Public and private keys

The solution is to encrypt the key for the message with the recipient's public key. When they receive the message, they can then decrypt the key for the message, and then decrypt the message itself. This is a bit convoluted, but it's much less processor-intensive than encrypting the whole message using public key encryption.

You can use OpenPGP in most mail clients, but we'll look at doing it in webmail. Since OpenPGP is purely a text format, you could generate the encrypted message elsewhere and copy and paste it into your email. That's exactly what we'll do, but instead of copy and paste, we'll use a browser extension to convert the plaintext to encrypted ciphertext.

Mailvelope (**www.mailvelope.com**) works with Chrome/Chromium and Firefox, and it comes pre-configured to work with some of the most popular webmail providers (Gmail, Yahoo, Outlook.com and GMX). Installing it is no more challenging than downloading the extension from its Releases section (**https://github.com/toberndo/mailvelope/releases**) and opening the file with the appropriate web browser.

The first step is to generate a public/secret key pair. In Chrome/Chromium, you can get to this by clicking on the padlock icon that should have appeared to the right of the address bar. In Firefox, this options menu is a little more hidden. First, you'll need to go to view

USING OTHER MAIL CLIENTS

We've described the process for working with Mailvelope, but the process is almost identical for all OpenPGP-compliant software. You shouldn't have any problems following along using Thunderbird or Evolution, or even AGP and K9 for Android or Cyanogenmod.

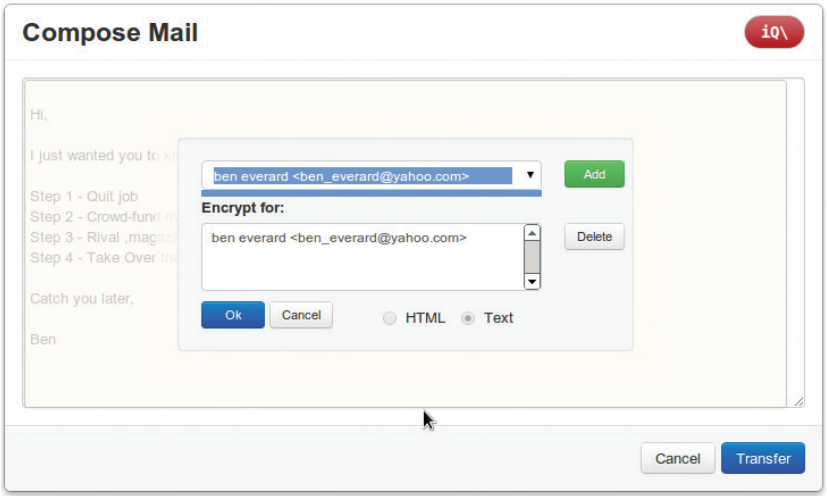
Regardless of the software, you'll still have to go through the same process of generating and exchanging keys before you can communicate with someone. As mentioned in the main text, you should be able to transfer keys between these pieces of software so you can access the same mail account through different programs.

Mailpile is a mail client designed to bring PGP to the masses by making it easier to set up OpenPGP encryption, even for new users. The project raised just over $163,000 in crowdfunding and is currently in development, and you can track its progress at **www.mailpile.is**.

DIGITAL SIGNING

OpenPGP encryption ensures that only the intended recipient can read the message; however, it doesn't guarantee that they receive the message, or prove who sent the message. Encryption can't help with the first of these, but there is something you can do about the latter measure.

In many OpenPGP mail clients (and the **gpg** command line tool), you can add a digital signature to a clear-text message. It does this by leaving the message in plain text, but also encrypting a hash of the message with your secret key. This encrypted hash is known as a digital signature. Since it's encrypted with your secret key, it can be decrypted with your public key. Any recipient that knows your public key can then decrypt this hash and check it against the message. If they match, the recipient knows that it really came from you.



You can send encrypted messages to several people at once, and Mailvelope encrypts it for each of them.

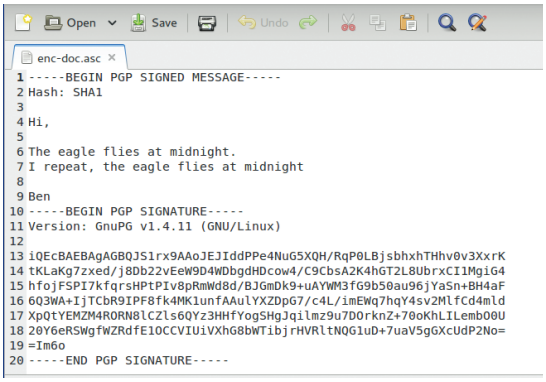
> Toolbars > Add-on bar. This will make the Add-on bar appear at the bottom of the screen, and then you should find the padlock icon on the right-hand side of this. This icon will bring up a menu, and you'll need to select Options (see the image, left).

In the Options screen, you can create a new public/secret (private) key pair by selecting Generate Keys. Once you've done this, you can go to the Display Keys screen to see it. This screen will show all the keys that Mailvelope knows, whether they're other people's public keys or your own public/secret key pairs.

Before you can receive emails, you have to send your key to the people you want to communicate with. The key file can be exported from the Display Keys screen (you can also export your public/private key pair here and import them into another mail program).

Getting the public key to the recipient can be a challenge. The best way to do this is to physically transport the key, as you can be completely sure that they got it correctly. The easiest way is just to email them the keyfile. However, it's possible for some malicious attacker to intercept this message and change the keyfile.

There are two other options: key servers and webs of trust. Key servers are databases of keys that you can add your keys to, and retrieve other people's keys from. For example, try **http://keyserver.pgp.com**



You can use gpg to create signed documents from the command line. Just run gpg --clear-sign <text-file> to generate a file containing the plain text and a signature.

or **http://pgp.mit.edu**. Of course, it is possible that some attacker could take control of one or more of these key servers and put fake keys in them. Webs of trust have a decentralised method of verifying keys. It's done by people digitally signing the keys of people they've met and exchanged keys with. If you need to communicate with someone, you can then tap into this web of trust and see who trusts them. Perhaps someone you trust also trusts them. Perhaps someone you trust trusts someone who trusts them. If this chain is short enough, then you can be confident that you can trust the person. Unfortunately, Mailvelope doesn't currently support webs of trust.

Keep it secret, keep it safe

As is so often the case, the decision on which way to distribute your key comes down to security versus convenience. If you're concerned, you could always follow up with another method such as a phone call to confirm the key. Once someone has sent you their key, you just need to load it into Mailvelope using the Import Keys screen in the Options.

Getting set up with keys is the hardest (or at least, most inconvenient) part of using any OpenPGP-based communication. Once you've done this, it's easy. With the Mailvelope extension running, just use your mail provider's web page as normal (if your mail provider isn't already on the Mailvelope watch list, you'll need to add it in the Options). When you get to the compose page, you'll see a floating icon of a pen and paper. Click on this and it will open a new window to let you enter the text for the message. Once you've written the message, click on the padlock, and add one or more people to the list that it's encrypted for, then Transfer to put the ciphertext into the email.

If you receive an encrypted message, Mailvelope will display a decrypt icon; click on this to enter the passphrase you entered when you generated the key. This password gives you some security even if an attacker gets access to your machine.

Provided you exchange keys securely, and keep your keys safe, OpenPGP provides security that is thought to be unbreakable with current technology.

LINUX VOICE
TUTORIAL

BREW PERFECT BEER WITH HELP FROM THE RASPBERRY PI

GRAHAM MORRISON

We love beer, we love the Raspberry Pi and we love the Arduino – so we're bringing them together for one awesome project.

7 STEPS TO BEER

- Brewing
- Cooling
- Fermenting
- Priming
- Bottling
- Ageing
- Drinking

GENERAL LINUX

HARDWARE

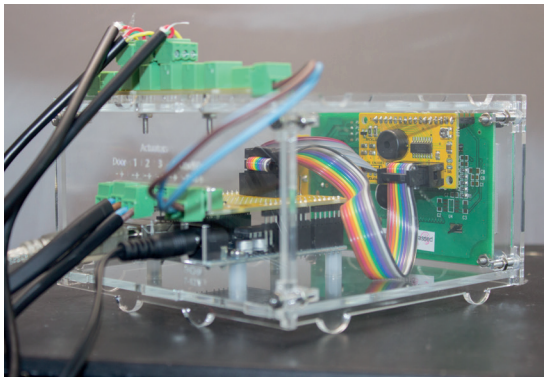
DIFFICULTY

DISCLAIMER

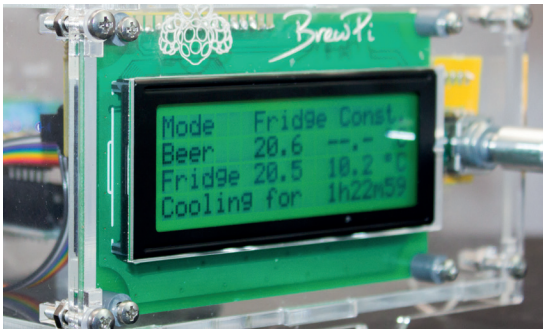
The following tutorial mixes liquid, electricity and DIY modifications, all of which can create a lethal cocktail of danger. Don't make any modifications yourself unless you're certain they're safe, and get a qualified electrician to check any modifications you do make.

Beer is lovely. But when you're making it at home, the biggest challenge (after discovering a way to boil vast quantities of water) is always finding somewhere to leave your brew to ferment. It's this stage of beer-making magic that turns what's known as wort into beer, creating alcohol and oodles of flavour. And for this stage to work well, you ideally need to be able to manage the temperature of the environment your beer is sitting in. In the UK, many amateur brewers resort to using an 'airing cupboard', normally situated next to the hot water tank and used for drying clothes. This isn't a bad place, because it's warmish – many beer kits like to ferment at around 20 degree centigrade – and the temperature doesn't fluctuate massively. But it still fluctuates, and it may even prove too warm. Many yeasts, especially for ale, prefer things a little cooler (18–20 degrees, ideally, but this depends on the beer). And lifting 25 litres of wort into a first-floor cupboard could break your back, and you've got a hygiene nightmare if it falls over, or falls through the flimsy shelf its sitting on.

BrewPi is the answer to this conundrum. It's a brilliant project that brings together a love of Linux, a little hardware hacking and plenty of beer into one fermenting barrel of hoppy goodness. It's essentially a device that controls the environment surrounding the fermenting bucket of beer, enabling you to make perfect beer every time, regardless of climate and house heating cycles. Many people use an old fridge or freezer as the surrounding container and connect the BrewPi to a cooling and heating mechanism to enable its clever algorithms to create the perfect environment for your beer. The BrewPi itself is a mixture of hardware, software and initiative. Not only has its creator, Elco Jacobs, built an incredibly effective system for fermenting beer, he's created an extremely helpful community of BrewPi enthusiasts, an online shop and an assembly system for easy access to all of the bits and pieces you'll need.



This shows the rear of the LCD connecting to the Arduino and the shield, with the OneWire connector above.



The various bits of the BrewPi give little indication that they can be put together to create something awesome.

What you'll need

While you will need a fair bit of kit, it needn't cost very much. The fridge or freezer is the biggest consideration, as well as somewhere to put it. We asked the internet, and Mark Einon in Wales very generously obliged with a freezer he was going to give to the local freecycle initiative (thanks Mark!) Almost any fridge or freezer will do, as long as it's working, and you should be able to find someone willing to let an old model go for very little. You need enough space within the freezer to stand your fermenting bin, and as our freezer's shelves were made from coolant pipes, we had to bend these back before there was enough room. Fortunately, the pipes were easily pushed back. We then slotted in an old wooden shelf to stand the fermenting bucket on, as they can be very heavy when full of 25 litres of brewing beer.

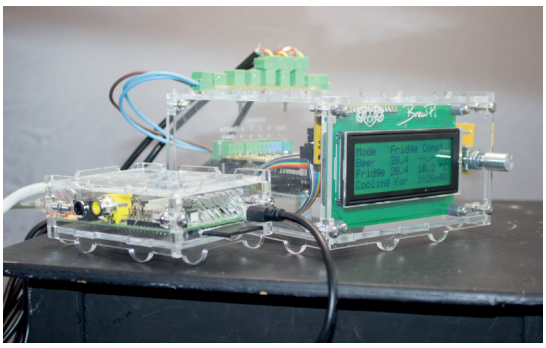
If the fridge or freezer has an inside light, this can be coerced into another essential task – heating up the inside environment. If not, you'll need some other kind of heating mechanism. Some people use a reptile mat wrapped around the fermenting bin, but we plumped for a 60W waterproof greenhouse heating bar, which cost us £15 new on eBay, and slotted nicely into the bottom of the freezer with plenty of room. You will also need both a Raspberry Pi, complete with a > 2GB SD card, and either an Arduino Duo or an Arduino Leonardo microcontroller. If you're anything like us, you've got an old Duo tucked away in a drawer

somewhere and a Raspberry Pi going spare. And despite the name of the project, there's no specific reason for requiring a Raspberry Pi – any Linux device with a USB port capable of running the Apache web server and some Python scripts should be up to the job. You might want to try a NAS, for example, if you're running one already. But the Pi is well suited to being tucked away in the garage, and it's relatively cheap, so it's still a great option. Most of the hard work is done by the Arduino, as this interfaces with the various sensors and relays and runs the complex controlling algorithms that adjust the temperatures within your freezer. Your brew will even keep brewing if the Pi crashes, which is handy if there's a power failure and your Pi develops a read/write error. The Pi is really just logging and serving up the data for the web portal.

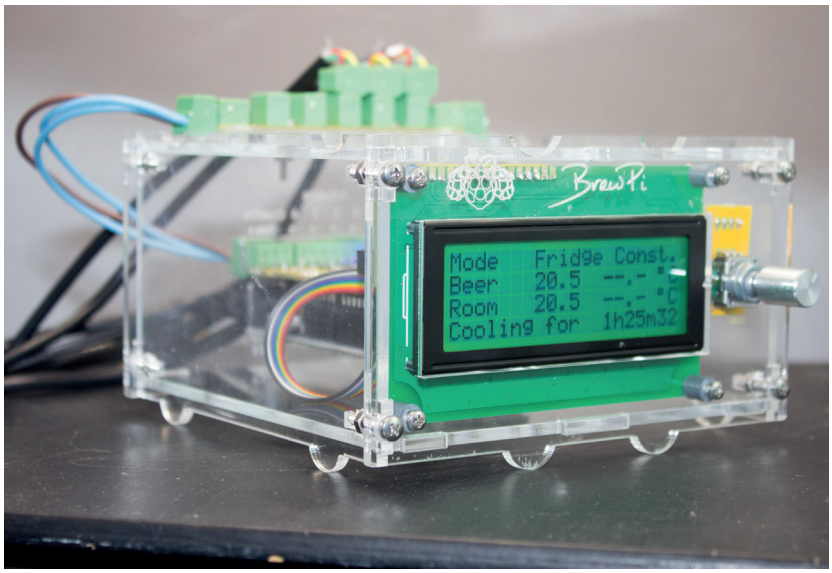
Unless you're an expert who's happy building circuits, you'll also need the BrewPi kit (**brewpi.com**). This includes everything you need to turn your Arduino into a sensor-wielding beer factory. It includes the shield, a PCB that slots onto the two compatible Arduino form factors, along with the LCD, the sensors, the actuators (more details later if none of this makes sense) and the other fiddly bits that may otherwise take an afternoon to source. It's even possible to buy the whole thing pre-constructed, but we think that's missing half the fun, especially when the build itself isn't that difficult.

We'd also highly recommend buying the case kits. These lasered bits of plastic encase both your Raspberry Pi and your Arduino to create a sleek, professional solution that looks great sitting atop your freezer. They also stop bits getting bashed about or falling off. Expect to pay around £70 for the shield and case kits together. You'll also need a miscellany of common tools to put the whole thing together; a soldering iron and solder, maybe a solder sucker, some tweezers, a range of differently sized screwdrivers and a steady hand.

Did we just say soldering iron? Yes! You'll need to solder the various components on to the Arduino shield. But it's straightforward, and this should make an ideal first project if you've not done any soldering before. All the components are large and there's no fiddly soldering required. Try watching a couple of YouTube soldering videos to familiarise yourself with



The Raspberry Pi can also fit on top of the BrewPi case, in a separate box or *au naturel*. Cases are good.



The BrewPi isn't an easier way of making beer. It's an easier way to make it perfect.

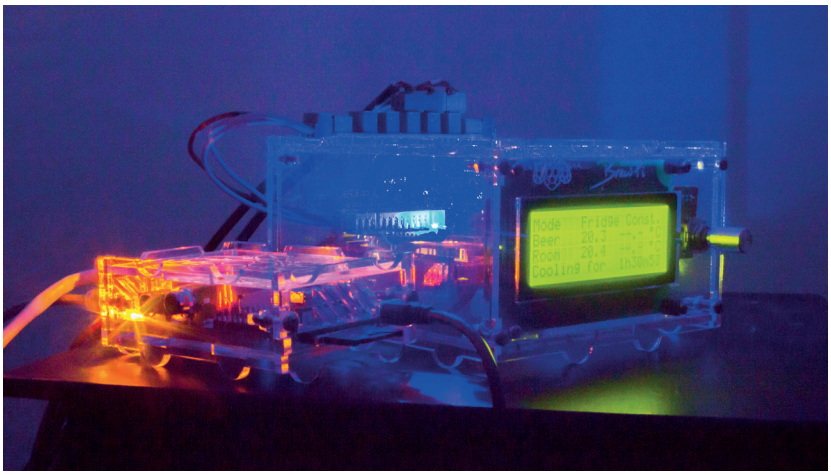
the process first, and then experiment a little with an old circuit board and some wire. You'll then be set for the main event.

The shield is the bit that attaches to the Arduino, and it's probably the most complex part of the whole assembly, so let's get this out of the way first. The main instructions can be found at **www.brewpi.com/brewpi-soldering-guide**, but we're going to cover the broad detail of the process, along with any particular notes we make along the way. The official instructions are made up of photos, and while they're great if you know what you're doing, we want to make the project as accessible as possible by making fewer assumptions about the builder than the official site.

Forging the shield

First, lay out all the components on a table top, grouping them together so you can check they're all there. This also makes it easier to install. Now start by being brave – you've got to snap the shield apart into four separate boards. It's a little like breaking bonfire toffee. The large board that breaks off (labelled with **www.brewpi.com**) connects directly to the Arduino. Then there's a long strip embedding seven columns of three holes, a medium-sized rectangle of a board with a surface mounted integrated circuit, and a tiny rectangle that will host the rotary encoder.

Break off the broken tabs remaining on the boards with a pair of pliers or a small pair of cable cutters so that the edges are as smooth as possible. Some of the pin arrays – the ones with the two collars of black plastic – are designed to fit on to your Arduino board so that it can connect to the holes on the shield. There are five of them, and you should find there's one for every header on the Arduino. These need to be connected to the Arduino first, before being soldered into the shield – this locks their orientation and connection. The longer pin goes into the Arduino, while the shorter piece goes into the shield. As we were using an ancient Arduino Uno, there were fewer power headers on the circuit board that pins allocated,



Red or blue LEDs on the shield indicate whether the BrewPi is currently heating or cooling your brew.

but the eight-pin array still fitted over the power pins and the 10-pin header still fitted across the IO pins without getting in the way of everything coming together. Don't forget there's also smaller six-pin rectangular connector. Fortunately, the shield only fits one way. Start your soldering at the corners to make sure all the pins stay aligned.

Now solder the single green connector onto the ACT1—ACT 4 shield holes, with the component attached to the side with the website URL. Connect a three-pin green connector to one side, and one of the two-pin connectors to the other (they all offer ports at right angles to the board, and have the same connector form factor as the eight-pin one you've just connected). Ours wobbled slightly while fitting them, so it's best to solder one of the middle pins first and wiggle the connector into alignment, before soldering any remaining pins. Flip the shield over and solder one of the 10-pin block connectors to the header labelled "To the LCD backpack", and make sure you've got the gap in the right place (facing the edge).

That's all that needs to be done to the main board! Congratulations. Now might be a good time for a cup of tea before moving on to the LCD backpack itself.

LV PRO TIP

Soldering tips; heat up the destination first, dab the solder onto the joint, make sure it flows into the joint naturally and try not to bridge any connections. If you do, heat and remove using a solder sucker.

Glowing electronic display

The LCD board is the one with the small integrated circuit already on it. The circular speaker fits into the middle with the upwards side on the same side as the chip, and after soldering, you need to cut the protruding pins from the other side. Another 10-pin header comes next, with the gap facing the integrated circuit. Flip this small board over (to the side without any components), and fit the 16-pin header into the holes. Solder from the other side.

The tiny board for the rotary encoder is up next. The official instructions mention that the biggest two pins on the encoder need to be squeezed slightly to fit into the holes. We didn't need to do this, but we did need to use a fair amount of strength to get the encoder into position. Make sure the side with the handle is the one with the circle on the board, and solder the joints from the other side. A washer, a nut and then the handle can be slipped over the encoder when you've finished.

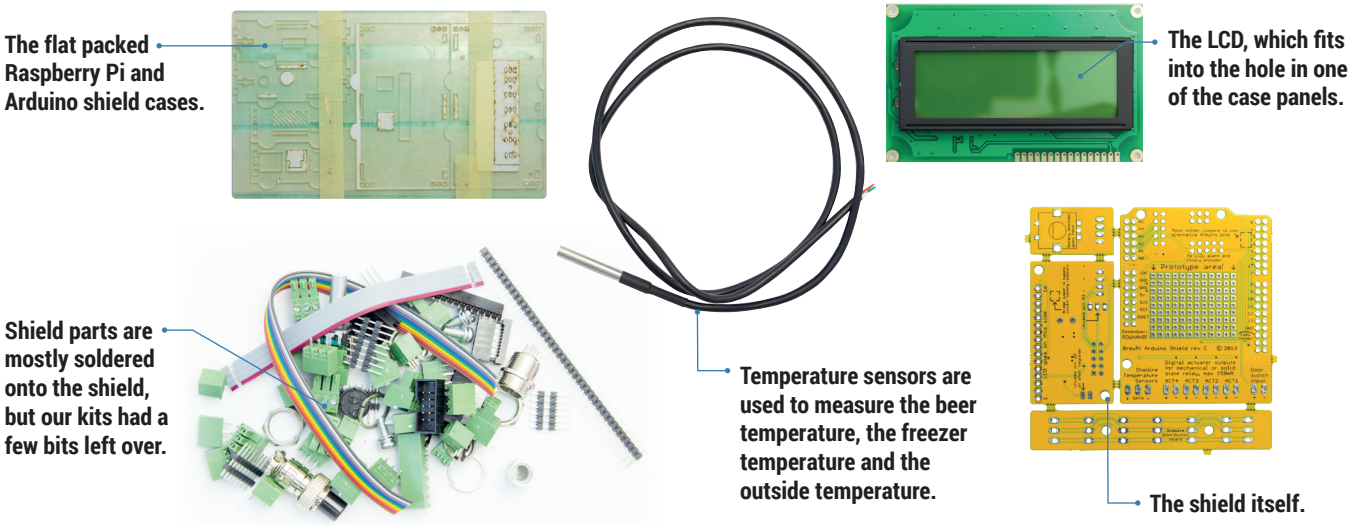
Next is what's known as the OneWire distribution board (the only board remaining). Sometimes it's written as '1-Wire', and it's a standard protocol for communicating with devices from Dallas Semiconductor (such as the temperature sensors we need for our BrewPi), using a single connector, hence its name. This needs seven of the three-pin green connectors — two shaped at right angles for the edge connectors, and the other five directly pointing up (you can see this illustrated on the board itself now you know what to look for, and that's the side they need to be connected to). Official instructions suggest starting with the two outer connectors, as these are oriented outwards lengthways. The other five all face upwards with their pins on the left when you're looking at the text on the board. The green 'AT-AT' connectors (for that is what they look like, not an official designation) then plug into these and the two end connectors.

Now it's the turn of the rainbow-coloured ribbon cable, which we need to turn into something a little more civilised to enable it to connect to the ports we've been soldering. If you've ever made your own IDE cable for an ancient PC, this is very similar. The black plastic connectors that attach themselves to the ribbon cable have teeth that penetrate the insulation on the outside of the wire to make a connection without soldering anything. Just make sure the triangles on the connector align with the black wire in the flat cable. Push the cable through until it just protrudes from the other side, and taking the advice of the official instructions again, place the smaller edge on a table and use something flat to put considerable pressure onto the connector. It should just about come together, and in so doing, connect the pins to the cable. When this seems secure, fold the long end of the cable up and over the back of the connector before sliding the remaining black connector to hold the cable together. This needs to be done on both sides of the ribbon cable, and both connectors need to point the same way so that the cable won't twist. That last bit can be a little mind bending as you try to work



We had to bend one of the shelves in our freezer to make enough room for the fermenting bin.

THE BREWPI SURVIVAL KIT



out which way to put the connector on so that the black cables stay in the same place and the connector is pointing in the same direction after you've twisted the cable back over the connector. You can now connect both of the boards with the correctly sized connector together with the cable, and we felt slightly more optimistic after testing the continuity of the connections to make sure we'd pushed through the connectors to the ribbon cable with enough pressure.

For the other ribbon cable, pull off the ends where they've been cut and wiggle this into the underside of the rotary controller board. Pin 4 should always be red. Then solder the pins to the board, The other end of this cable goes to the LCD board, parallel to the rainbow ribbon cable, and connected to the same side. Make sure pin 4 lines up and solder this as well.

The next stage is the LCD, and you first need to break off 16 pins for the LCD itself. The official guide has a great tip, where you connect the whole header to the female header on the other board and use this as a guide for snapping the 40-pin header at the right place with your hands. This didn't quite work for us, as we broke the header one pin short, but it was easy enough to solder the lone pin alongside the others. Solder these pins on the top surface (the same side as the LCD itself), and you can now attach the LCD to the female header.



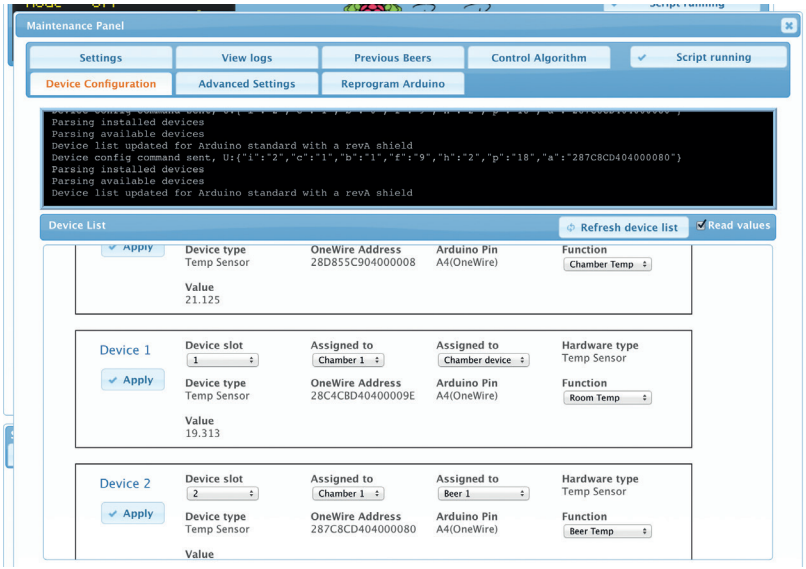
To make the sensors inside the fridge easily removeable, use a connector like this within a container.

The final stage of shield forging is to take the sensors and strip the insulation off the end of the wires – a couple of millimeters will do. Each cable has three 'cores', and each core needs to be screwed into a three headed 'AT-AT' green connector, so that when these plug into the OneWire board, red is at the top (marked 5V – this is important), and yellow at the bottom. The official instructions note that the colour order of the yellow and green wires has changed, so it's worth making doubly sure if you're reading this in the distant future, as the sensors might not be able to take 5V going in the wrong cable. To make the ends of the wires easier to insert into the tiny screw holes, and to make them more resilient, it's worth dabbing them in a little molten solder.

Porter, Stout, IPA – and the case

You now have a choice. You can either keep the OneWire connector close to the rest of your BrewPi hardware, or place it closer to where the sensors are going to be. This might be useful if you wanted to position the OneWire board within the fridge, for example, but we decided to go with the official instructions and wire up a short three-core cable (maybe 20cm), with AT-AT connectors at either end, to connect the OneWire board to the BrewPi. We used an old power cable with earth for easy access to three cores with insulation attached. This cable eventually loops outside the case from the main board to the OneWire connector.

The cases are all made from various bits of lasered plastic, and it's never clear exactly what goes where. It's like a BrewPi 3D jigsaw puzzle. The Raspberry Pi case is a good place to start, as this is emblazoned



You can check your sensor devices are working by enabling the 'Read values' option before refreshing the device list.

with the Raspberry Pi logo flanked by some hops, and it's also obvious which way the pieces should go when you attempt to fit your Pi into the case. The feet of all the cases are half-circles, which is another good way of orienting yourself with the 13 or more pieces used to construct each of the cases.

As we we're using an early Pi, lacking holes on the PCB, there's no way of mounting the board inside the case. The official instructions show a couple of spacers and screws mounting the Pi to the lower case panel. Our case design didn't have a hole even if we did want to connect the Pi. But thanks to the various prominent ports and connectors on the Pi, it was held firmly in place regardless. One side has the video and audio connectors, the opposite just an HDMI connector. Lengthways, theres a micro USB at one end and USB and Ethernet at the other. It's also a good idea to push out any of the small bits of plastic that are used to create airflow through the case, as the Pi can be prone to overheating, but we couldn't remove some of these pieces as they weren't separated enough from the borders of the plastic. This may have been why two extra end pieces, with all the bits removed, were hidden away in one of the part bags.

It all goes together easily enough when you've worked out up and down and where each side fits. Be careful with the side containing the HDMI connector, as it's not immediately obvious when it aligns and you may not notice it's reversed until the end. When you've got everything held together, you've got to now use the long screws, two at each long end, to go through a washer, then into the case, and then through a nut you hold in the small vertical gap before tightening the whole thing up. It's fiddly and frustrating, so we'd suggest focusing on the beer.

Construction time again

This leaves you with significantly fewer bits to worry about for the other case, which is going to contain our BrewPi shield. Now, for some reason, our case is a hybrid of an earlier revision with a few differences between both the earlier version and the 2.0 cases, so there's no point telling you how to put the case together. In fact, the 3.0 case was announced in January, and is smaller again. We were able to make it up as we went along because it's much easier than building the shield, and mostly common sense. There

POWERING THE BREWPI AND UPDATING THE FIRMWARE

Before we move on to software, you need to give some consideration to how you're going to power both the Raspberry Pi and the Arduino. In theory, you could power the Arduino from the Raspberry Pi's USB, using only a single hub or adaptor. We tried this with as many milliamps as we could muster, but the LCD on the Arduino still dimmed when we did anything. Rather than take any risks with our beer, we decided to power both separately. As we all know, the Raspberry Pi is very susceptible to irregularities in power, so it's best not to take any risks – use a high amperage USB hub or adaptor for the PI, and an appropriate adaptor for the Arduino.

It's now time to test whether your soldering skills have been good enough, and to stretch a few of those Linux skills too! The first step is to get a working Raspberry Pi configuration, complete with your chosen method of network connection. This has been documented many times, so we won't go into the details – plus, downloading and installing NOOBS onto your Raspberry Pi makes the whole process easier than ever. Just make sure the Raspbian installation and the firmware is up to date, because there are some known issues with Raspberry Pi stability, especially with older versions. And stability is key when you're asking a Raspberry Pi to control temperatures for a week or two.To update Raspbian, type:

```
sudo apt-get update
sudo apt-get upgrade
```

To update the firmware, type:

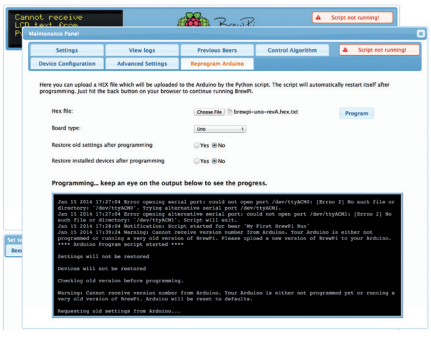
```
sudo apt-get install rpi-update
sudo rpi-update
```

We now need to grab the latest installation tools. To do that, just enter the following and leave all the answers at their default values:

```
git clone https://github.com/BrewPi/brewpi-tools.
git ~/brewpi-tools
sudo ~/brewpi-tools/install.sh
```

After this has completed, reboot your Pi. You will now be able to point a web browser on your LAN to the IP address of your BrewPi. Don't (yet) get distracted by the blinking lights, as they're not doing anything meaningful. Instead, you need to upload the BrewPi firmware to the Arduino before anything can happen. First download the firmware file itself (here's the link: **http://dl.brewpi.com/brewpi-avr/stable**), and make sure you get the correct file. The file depends on your Arduino type and revision – ours is an Arduino Uno Rev A, for instance. To upload this to your BrewPi, click on the 'Maintenance Panel' button on the right of the web interface, then click on 'Reprogram Arduino'.

Select your Arduino from the drop-down menu, then select the downloaded hex file. Make sure 'No' is answered for both the 'Restore Old Settings After Programming' and 'Restore Installed Devices After Programming' options and click on the 'Program' button. You'll see the output of what's happening in the black box below, but with a bit of luck, the BrewPi will beep a couple of times and a few minutes later, you'll have a programmed BrewPi.

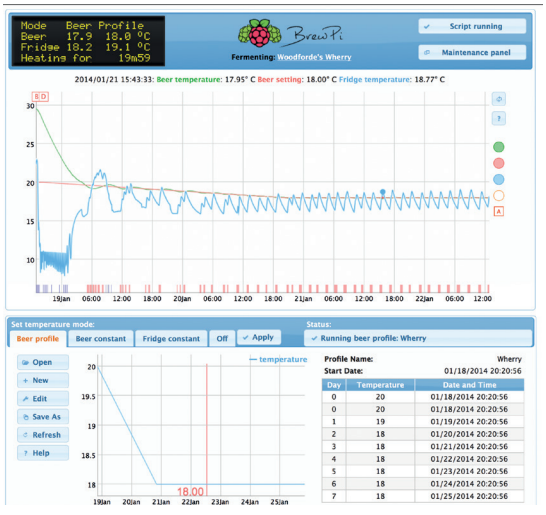


When you update the firmware of the BrewPi, the output console keeps you updated on progress. It only takes a couple of minutes.

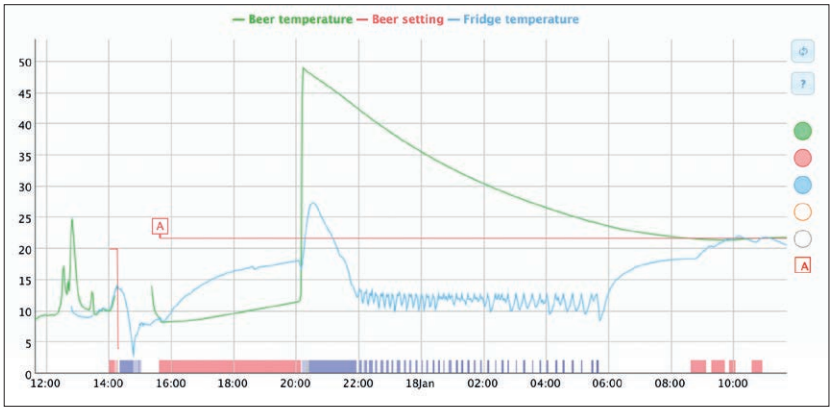
are three different kinds of bolt – two of identical length but slightly different widths, which you'll find out when you try to squeeze a larger one into the smaller holes, but you might notice the other way around, so it's still worth laying everything out before you start, Similarly, there are two different kinds of nut, although on first glance they all look identical, and the case building consists of two separate small phases – connecting the Arduino to the case followed by the LCD panel we built into the shield earlier. The grey threadless spacers are used to distance the LCD from the edge of the case, while the threaded white spacers are used for the Arduino. The position of the holes through the Arduino PCB mean that it can only be fitted onto the case one way – with the power and USB connector along the rear edge.

As we mentioned earlier, you also have the choice of whether to mount the OneWire board to the top panel or mount this inside your freezer cabinet so that the sensors plug directly into this within the freezer. As we opted to mount it to the case, and you need to use the provided small plastic panel (with OneWire embossed onto its top surface, along with numbers for each input). Two of the narrow bolts go through the PCB, through the small plastic panel, through the case, through a washer and finally onto a nut to make this happen.

After connecting the Arduino to the case and making a decision about the OneWire connection, we now need to put everything together like a simple 3D jigsaw puzzle. The half-circle plastic nodules are the feet, and to get ours together, we first fitted the rear panel. This is the one with the holes for power, USB and the controller connectors, and after you've placed it over the Arduino ports, you can hold it in by plugging in the green 'AT-AT' connectors to the outside of the case. They fit in pairs with the exception of the single three-pin connection on one edge. The two side panels then slid into the rear panel, followed by the top and finally the LCD, which slid onto those to all of the other panels to make the front. Don't forget that many



Our first brew started at 20 degrees and lowed to 18 after 48 hours, to create the best temperature for the beer.



The BrewPi is brilliant at controlling temperature. Here's the sensor output after we put a bin of 50°C water into the fridge and asked the BrewPi to take the temperature down to 21°C.

of these panels have a thin layer of plastic that can be removed, along with a few squares for the joints that may not have fallen out with the laser cutting.

Eight of the remaining screws now pull the case together, in the same way that they did for the Raspberry Pi case. The official instructions suggest using a magnet to hold the nut in place, but we we found it easier to push the bolt in until it reaches the gap for the nut, then ease the nut into place using the nut to make sure it doesn't go too far and drop inside the case (which is going to happen with the last one anyway – stay calm and think of beer). A quick tip if one does fall in, you can play an amusing game with yourself and attempt to bounce the nut back out of the same hole - it's not that difficult but looks a little deranged. Sensible people will loosen the bolts at one end to separate the box enough, which is also a good way of taking the top of the case without removing any of the bolts. And don't forget the washers on the outside. They're needed to make the bolt fit.

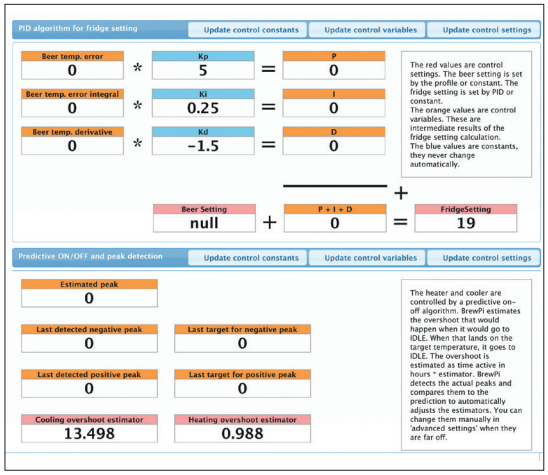
Loose fit

But we'd suggest maybe loosely taping the case together for now, until you've been able to test out your BrewPi with the software to ensure that everything works. That way you don't get doubly frustrated by something not working and having to go through the whole unscrewing process again. You now need to connect the two SSR blocks to the outputs on the shield, making sure you get the positive cable going to the positive input and the negative cable going t the negative input on the SSR. These solid state relays perform a simple job, turning the power going through the other two points either on or off. This is used by the BrewPi to automatically turn on refrigeration or heating. Some BrewPiers have reverse engineered their refrigeration units and heaters to splice these connections into the most efficient place. We cut open the power cables to both the freezer and the heater, took out and cut the negative wire, and used this on other side of the power output on both SSRs. The power output was on the top of our SSRs, while the control inputs were in the bottom. Make sure you get this correct and that your wiring is safe, because you could easily create a hazard at this step. You should also consider the

LV PRO TIP

Although not essential, a cheap multimeter can make testing much easier – especially if it makes a sound when the two contacts connect. This is called testing for continuity, and it's a great way to make sure dodgy soldering is working.

The algorithm that controls the BrewPi is complex, but you can even fine tune this from the Maintenance panel if you so desire.



location of the SSRs, as they're usually exposed and obviously shouldn't go anywhere near liquid.

Back on the BrewPi shield, one output to the SSR triggers a red LED while the other triggers a blue LED, so it's worth getting them correctly connected as you can then see when your device is heating or cooling. These connections are on the backside of the shield, not on the OneWire connector – that's just used for the sensors at the moment, although there's talk of adding a hydrometer reader to measure the alcohol content, which is something we'd love to see.

Now stop. It's time to admire your work. The tough bit is over with, as the BrewPi is now built, waiting only for a little Linux magic to bring it life. And you know all those holes in the top of the BrewPi case? And the weird semi circle feet on the Raspberry Pi case? They fit together! Your Raspberry Pi should sit snugly to the top of the case like the Boeing 747 of brewing.

Configuring devices

The very final step (we promise!), is to tell your BrewPi exactly what you've got connected, and we found it easier to start with a blank canvas. Click on 'Device Configuration' button from the Maintenance panel and you'll see a list of devices your BrewPi thinks are connected. The devices are the switches to control the heating and cooling, plus the two or three sensors you've got connected. If any devices appear in the Installed Devices list, set their function (a drop-down list on the right of each entry) to 'None' and click Apply. This will move them from the 'Installed Devices' box to the 'Detected Devices' box, from where we can now add them as we need to. Enable 'Read Values' and click on Refresh Devices. Click on the 'Refresh Device List' button and enable the 'Read Values' check box. This will list connected devices along with a number to indicate what the switch or sensor is reading. You can easily detect and check your sensors are functioning in this way. OneWire works with unique identifiers embedded within each device, so the device ID is unique for each sensor, not for the BrewPi configuration. That means if you identify which sensor you're going to use within your fermenting bin, you can plus this into any of the OneWire connectors. We checked sensor was working by plugging each in turn and refreshing the device list to make sure a temperature value was being read. We also identified each sensor by heating or cooling the sensor and wrote down which one was which.

You need two sensors for the BrewPi to work properly. One measures the ambient temperature within your fridge or freezer, while the other measures the temperature within the beer. For the beer measurement, it's recommended you use a 'thermowell' to keep the sensor separate from your beer. You also need to solve the problem of getting the sensor cables into the fridge or freezer cavity. Some users piggyback their wires onto any wires they can already find going into fridge. Our approach was to butcher an Ethernet cable – there are more than enough cores within one of these for 2 of the sensors – and drill a tight-fitting hole for both this cable and the power cable for the heating unit, into the side of the freezer. This has worked with no problems so far, and not affected the insulation of the freezer.

Brewing your first beer

With sensors in place and the software running on your BrewPi, you're ready to brew. Despite the slightly intimidating appearance of the web interface, it's very straightforward to use. Click on the 'fermenting' link just below the BrewPi logo and you'll be given the option of starting a new brew. You can do this to log the details of each brew, as well as clear the data for the start of a new fermentation cycle. The main display area is taken up by a graph showing the changes in beer temperature (green) and freezer temperature (blue), as well as the temperature outside the fridge, although this isn't used by its algorithms. At the bottom, along the timeline, blue and red blocks show when the cooling and heating was engaged.

There are three modes for fermenting your brew; Beer Constant, Fridge Constant and Beer Profile. Beer Constant simply keeps the beer at a specific temperature, which you dial into the large number bar at the bottom of the screen. Expanding on this, the Beer Profile setting enables you to set a desired beer temperature for each day. This is useful if you want to try a slightly warmer environment at the beginning and end of the fermenting cycle. When either of these beer profiles are active, the LCD display shows the absolute temperature as well as the temperature for the profile. This is the target temperature for the algorithm, and you'll find the BrewPi will cool or heat to nudge the temperature closer to the desired value.

The Fridge Constant setting does what it says, keeping the temperature of the fridge at a specific value. This might be useful for the couple of days after you've bottled your beer, or put it in a cask, as you usually have a couple of days of secondary fermentation. But it could be equally useful for cooling your final product for the final, essential step of brewing beer – keeping your home-brew ready to drink at a perfect temperature, all year round.

BREWING YOUR OWN BEER: A BRIEF ENCOUNTER

The world of homebrew will feel familiar – it's full of people who obsess over details and argue endlessly about packages.

Homebrew forums across the internet are full of enthusiasts arguing over every detail of the brewing process. And we mean every detail. Fermentation temperature is a dark art of its own, as is the amount of priming sugar to use – we've seen simpler algorithms explain Bézier curves in OpenGL!

As with Linux, all this data and debate can be totally overwhelming to the beginner. But again like Linux, it's worth struggling through to the other side. Just think of the beer.

We also see no shame in starting small. Beer kits are perfect for this. They can be a little pricey, but they'll take the pain out of your first brew. To get started, you'll need some simple pieces of kit. Here's what we recommend:

1 A 25-litre fermentation bin
This doesn't need to be absolutely airtight, as the brewing process will create CO_2, which sits on the top to create an airlock. We drilled a hole in the top to encase one of our BrewPi sensors within its own well.

2 A similar sized pressure barrel
The pressure part is important for the secondary fermentation preocess, because it's what carbonates your beer and keeps your beer fresh. We'd recommend a pressure valve with a connector for a CO_2 canister. These are relatively cheap, and they're used to create a CO_2 buffer when the pressure gets too low to push the beer out effectively. If you don't want to use a pressure barrel, you can use bottles with caps.



Just like open source software, you can create your own recipe or you can stand on the shoulders of giants. Image credit http://superflex.net

3 Sanitiser
Everything that comes into contact with your developmental beer has to be free of any harmful bacteria. Bacteria and wild yeast kill beer over the period it is stored, leading to feelings similar to a hard drive failure.

4 A syphon and hydrometer.
The syphon is to transfer your beer from the fermentation bin to the pressure barrel or bottles, while the hydrometer is to calculate how much alcohol is in your brew. You must measure the gravity at the beginning and the end of the process for this to work – taking a measurement at the end isn't enough.

The biggest threats to your beer are sanitisation, as we've already mentioned, and temperature fluctuation, which is solved with the BrewPi. Another tip we've found helpful is to cover all threads (such as those for the tap, the top and the valve on the pressure barrel) with Vaseline, as this helps to keep them airtight.

After you've whetted your appetite with a beer kit or two, it's time to move up to replacing the kit with your own. There are thousands of years of experience on the subject, and to be honest, we've only just started. But a good place to look for your first brew is a recipe that is itself open source.

Free Beer

This is exactly what is offered at **FreeBeer.org**, a tested and refined recipe for making excellent beer that's been released CC-BY-SA.

The ingredients list five different types of malt, Guaraná beans for added spice and energy and London ale yeast. This is followed by step-by-step instructions that will take your beer from mash to wort to fermentation to beer in as little as three weeks, all in the name of Free Beer. If you do get around to making some, and you have a bottle left over, you know where to send them.

LINUX VOICE TUTORIAL

OWNCLOUD 6: RUN YOUR OWN CLOUD

MIKE SAUNDERS

Love having your data in the cloud? Scared that the NSA and GHQC are tapping it all? Then it's time to set up your own server!

WHY DO THIS?

- Share and sync files, contacts and calendars across all your devices
- Collaborate on shared ODF documents with multiple users
- Keep your data safe from the prying eyes of big businesses

We're not big fans of buzzwords at Linux Voice. We don't leverage synergies, we don't harness data silos, and we most certainly don't streamline our paradigms. At first, the term "cloud computing" came under this umbrella of linguistic silliness, because it basically meant "doing stuff on someone else's computers", like many people have already been doing for years. But over time the term has become widely accepted, so we'll grudgingly use it. Bah humbug!

Now, there are many providers of cloud-like services on the net. DropBox, for instance, provides data storage and file sharing, while Google's ever-growing range of services includes document collaboration (Google Drive) and calendars. Many of these third-party services are packed with features and are easy to use, but they all have one problem in common: they all have access to your data. If your files consist of nothing more than lolcat pictures, and your calendar is simply used to plan your pub visits, you're probably not concerned about this. But if you're storing sensitive information – such as business plans – then it's wise to be cautious, especially in the wake of the Snowden revelations and US constitution-burning, NSA-spying shenanigans.

Do it yourself

One way around this is to host your own cloud services. It sounds like a contradiction in terms: isn't the point of "cloud computing" that you offload all the work to someone else? Well, yes, but by hosting your own cloud you can still have some of the benefits, such as sharing data and providing collaboration services across multiple machines and users. You control the hardware and software, and determine who accesses your data, but you still have the convenience of cloud-like facilities.

Arguably the best open source cloud package at the moment is ownCloud, which reached version 6 in December. It's loaded with useful features for file storage, file sharing, calendars and document collaboration, all accessible through a web browser, so here we'll show you how to set it up and explore the goodies contained therein.

1 GET IT INSTALLED

You can install ownCloud on wide range of distros, and if you're just playing around to learn the software, it doesn't matter if you're using a rapidly changing, cutting-edge distro – Arch Linux, for example. If you're planning to use ownCloud for real work, however, we recommend using a highly stable and long-term supported distro such as Debian or CentOS – we'll be using Debian 7.3 in this tutorial.

OwnCloud is written in PHP and can use a variety of web servers and databases. For simplicity's sake, we'll be using the well-known Apache web server here, along with SQLite to store metadata for the files. This is perfectly fine for a typical setup; if you end up really hammering your ownCloud server, though, you may want to switch to a more lightweight web server (such as Nginx) and a full-on database such as MySQL for extra performance.

To get the dependencies on Debian 7.3, use the following command:

```
apt-get install apache2 php5 libapache2-mod-php5 php5-sqlite
php5-common php5-gd php-xml-parser php5-intl php5-mcrypt
php5-curl ntp curl php5-imagick php-apc
```

Along with Apache, PHP and SQLite, this also adds some extras for generating thumbnail images of files and speeding up PHP scripts. Once these packages are installed, Apache should be started automatically – you can check that Apache is running by accessing the IP address of the Apache server in your browser (or going to **http://127.0.0.1** if you've installed it on your local machine).

Tarball time

Next, grab the **.tar.bz2** file of the latest ownCloud release from **www.owncloud.org**. At the time of writing, this was **owncloud-6.0.0a.tar.bz2**, but by the time you read this a newer version may be available. If so, just replace the version number accordingly in the command below. Extract the archive into your web server's document directory, eg:

```
cd /var/www/
tar xfv /path/to/owncloud-6.0.0a.tar.bz2
```

A bare installation takes up 155MB. We're almost ready to start using ownCloud now, but beforehand we have to make a few tweaks. First, we need to create a "data" directory inside the ownCloud installation, and make it (along with the "apps" and

"config" directories and the **.htaccess** file) writeable by the web server, which uses the "www-data" account in Debian:

```
mkdir owncloud/data
cd owncloud
chown -R www-data:www-data data apps config .htaccess
```

Note that some other distros use different user accounts to "www-data" for Apache, such as "http" or "apache". To find this out, run **ps aux** and look for the **apache2/httpd** processes, and then the username in the first column from the output.

Now we need to add some extra options to the Apache configuration file, which in Debian is **/etc/apache2/apache2.conf**. If you're using Apache 2.2 (the default in Debian 7.3) then add this to the bottom of the file:

```
<Directory /var/www/owncloud>
  Options Indexes FollowSymLinks MultiViews
  AllowOverride All
  Order allow,deny
  allow from all
</Directory>
```

If you've installed ownCloud in a different directory, change the path in the first line. And for Apache 2.4 systems, you'll need to change the "allow from all" line to read "Require all granted" instead. Once you've made the changes, enable URL rewriting and restart Apache with the following commands:

```
a2enmod rewrite
service apache2 restart
```

(For distros using systemd, try **systemctl restart httpd.service** to restart Apache.)

That's it – all the command line preparation is done now. Access the server in your web browser (eg **http://127.0.0.1/owncloud/** if it's installed on your local machine) and you should see the ownCloud login screen, as per the screenshot above.



If all has gone smoothly with the Apache setup, you'll see this screen when you first browse to the server. Now the fun begins...

2 SET IT UP

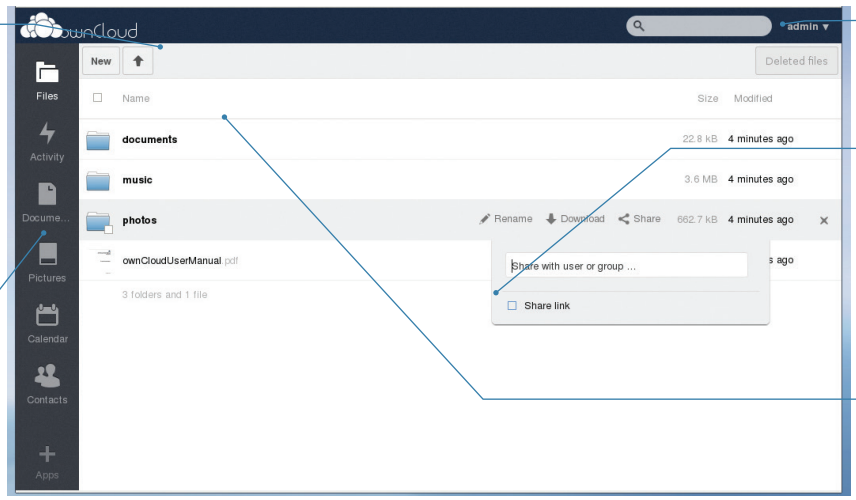
The first thing you'll need to do is create an admin username and password. ownCloud will do some background work, setting up its database, and you'll be dropped into the main screen. A pop-up will point you to a selection of desktop and mobile apps you can use to access your ownCloud installation. If you have a smartphone, it's worth trying these out.

A good way to understand the relationship between the different features is to click on the "photos" directory and then the up arrow, and upload a random image from your collection. If you now go click the Pictures icon on the left, you'll see your newly uploaded image, albeit presented in a much more attractive manner than in the normal file manager.

NAVIGATING OWNCLOUD 6

Create
Click New to create a new folder or text file, or the up arrow button to upload a file from your computer into the current directory.

Features
These icons switch between the different features provided by ownCloud, such as the document editor and contacts list.



User menu
Click here to change settings or log out.

File options
Hover the mouse over a folder or file, and you'll be presented with extra options to rename, share or download. Click the X button to delete.

Browse
Click on folders to open them, and on files to preview them.

While you're here, click on Activity on the left and you'll see a list of changes to your files. If you go back into the Files view and delete your image (via the X button), you'll see a "Deleted files" button appear in the top-right, from which you can restore files to their original locations.

Performance tuning

Periodically, ownCloud needs to execute some background jobs to manage its database and keep things running smoothly. By default this background job is run every time you load a page in ownCloud, but this impacts performance – it's much better to do it via a Cron job. In Debian, run **crontab -u www-data -e** as root, and then add this line to the bottom of the file:

```
*/15  *  *  *  * php -f /var/www/owncloud/cron.php
```

This runs ownCloud's **cron.php** script every 15 minutes. If your distro runs Apache under a different username, change it in the crontab command, and alter the path for **owncloud/cron.php** if you installed it in a different directory. Back in the ownCloud web interface, click the admin username (top-right), Admin, and scroll down to the Cron section. Make sure the Cron option is ticked (instead of AJAX or Webcron). By default, ownCloud's upload limit is set to 513MB (and potentially made even smaller by PHP's settings), which isn't very useful if you plan to use it for backups and video files. To fix this, go to the "File handling" section of the Admin page in ownCloud, and update the number to something more flexible (eg 8GB). You'll also need to change PHP's settings as well – so edit **/etc/php5/apache2/php.ini**, changing these lines:

```
upload_max_filesize
post_max_size
output_buffering
```

For the first two, set them to "8G", and for the last one use "8192". Restart Apache (as described earlier) and you'll be able to upload much larger files.

What's new in ownCloud 6?

If you've been running ownCloud for a while, and you're still using version 5, it's well worth upgrading to the latest release. Major new features include:

- **ownCloud Documents** Edit rich text documents with other users. It's not as featureful as Google Drive just yet, but it's a major boost for ownCloud and takes it way beyond just storage and calendars. The back end uses ODF, the same file format used by Open/LibreOffice, so you can easily export your documents for local editing.
- **User avatars** User accounts can now be accompanied by pictures. While this isn't a massive productivity boost, seeing images and not just names makes the interface nicer to work with.
- **Activities** A new view shows you recent activity in your account, such as changes to files.
- **Better conflict handling** Previous ownCloud releases were a bit rubbish if you tried to upload a file that already existed, but you can now choose to replace or rename a file when you're uploading.

3 AND EXPLORE IT!

It's not a good idea to use the administrator account for day-to-day work, so click on the Admin button in the top-right and then Users from the menu. Here you can type in a login name and password, and click the Create button to add the user to the database. If needed, you can also limit the amount of storage space allocated to the account.

So, click Admin > Log Out, and then log in with your normal user account. You're now ready to start exploring ownCloud's features in depth. You already have a bit of experience with the Files view: it acts as a simple file manager, and is a good way to organise your files so that you can access them from any machine on your network via a web browser.

But wouldn't it be better if you could access ownCloud data in a proper desktop file manager? Well, that's possible thanks to ownCloud's WebDAV support. In Gnome 3's Nautilus file manager, click Files > Connect To Server and enter the following:

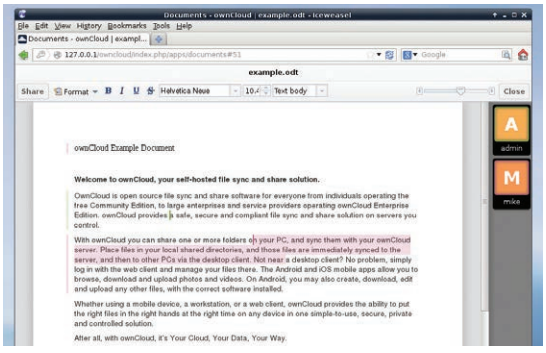
```
dav://127.0.0.1/owncloud/remote.php/webdav
```

Replace the IP address if necessary, and if you've enabled SSL support (as per the boxout), change "dav" to "davs" here. Xfce users can browse ownCloud shares in Thunar by clicking Go > Open Location and using the above address, while in KDE's Dolphin, click in the address area and enter:

```
webdav://127.0.0.1/owncloud/remote.php/webdav
```

After logging in with your ownCloud username and password, you'll be able to browse your files and upload new ones by dropping them into the window.

To share files with the outside world, hover over a file and click the Share button. You can either share the item with another user on the ownCloud installation, or generate a link (optionally password protected) to give to anyone on the internet.



We're logged in as "admin", and our changes are marked with light green. User "mike" is also logged in, and has selected some text marked with pink. Collaboration ahoy!

OwnCloud's calendar is simple, but useful: you can switch between day, week and month views, and click on an empty space to add an event. It's possible to set these events as all-dayers and make them repeat across multiple days. Under the Advanced button you'll find options for adding extra information such as a location, category and description.

If one calendar doesn't suffice for your work, click the cog (settings) icon in the top-right, just beneath your username. Here you can add extra calendars and also customise your time zone and time format. As with the Files view, you can also access your calendar from external apps: in the same settings panel, scroll down to the bottom where you'll see a URLs section. This provides you with CalDAV addresses that you can use with CalDAV-compatible apps such as Kontact and Evolution. Many mobile apps also support CalDAV, so you can keep your ownCloud calendar up to date when you're on the road.

In the Contacts view you can add entries and group them together. The cog button here also provides some useful features – for instance, a CardDAV URL that you can input into external contact management apps (click the globe icon). It's also possible to export your contacts list in **.vcf** format.

Documents

And here we come to the final big feature of ownCloud: document collaboration. This was introduced in version 6 (see the boxout, left), and while it's not especially useful for complicated documents at the moment, it does a decent job for basic rich-text editing jobs. When you click the Documents view, you'll see that a sample has already been provided for playing around with – **example.odt**.

Click on it and you'll see a minimalist word processor-esque interface, letting you add basic formatting to the text. But the most interesting part of this is the collaboration: start editing the text, and you'll see a coloured bar appear on the left-hand side, next to the paragraph that you modified. You'll notice that the colour of the bar matches the outline for your user icon on the right.

Click on Share in the top-left, and enter the name of another user (or the administrator, if you've only created one user account). Then, in a different web browser (so that you can have multiple sessions going), log into your ownCloud installation as that other user and go to the Documents view. You can now edit the document in both browser windows, seeing the changes that each user makes.

OwnCloud Documents is still in its infancy, but it already provides a great escape from Google Drive for many jobs, and it will just keep on getting better and better. If you love Google Drive's convenience but hate the thought of being spied on, why not give it a go?

Mike Saunders uses ROT13 encryption everywhere for maximum security. Abg ernyyl – ebg26 vf zhpu fnsre!

How to enable secure (SSL) connections

If you plan to access your ownCloud installation from the outside world, you'll want to enable SSL connections to encrypt your data as it travels across the internet. Here's how.

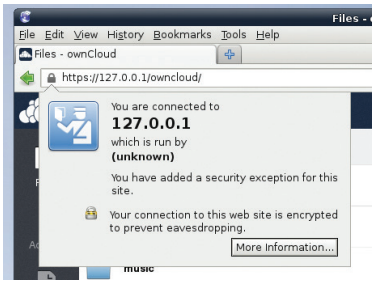
First, make sure that you have OpenSSL installed (**apt-get install openssl**) and enabled in Apache (**a2enmod ssl**). Then create a self-signed SSL certificate as follows:

```
mkdir /etc/apache2/ssl
openssl req -new -x509 -days 365 -nodes
-out /etc/apache2/ssl/owncloud.pem
-keyout /etc/apache2/ssl/owncloud.key
```

Now create /etc/apache2/conf.d/owncloud.conf with the following contents:

```
<VirtualHost 127.0.0.1:443>
SSLEngine on
SSLCertificateFile /etc/apache2/ssl/
owncloud.pem
SSLCertificateKeyFile /etc/apache2/ssl/
owncloud.key
DocumentRoot /var/www
<Directory owncloud>
AllowOverride All
order allow,deny
Allow from all
</Directory>
</VirtualHost>
```

If you're not testing ownCloud on your local machine, replace **127.0.0.1** in the first line with the IP address of the ownCloud server (you can discover this by running the **ifconfig** command on the server). And, of course, change the paths to the ownCloud installation where necessary. Restart Apache and access ownCloud via HTTPS, eg **https://127.0.0.1/owncloud/**.



You can tell your browser to accept the self-signed SSL certificate – it's safe.

ownCloud on the Raspberry Pi?

Yes, it's possible. And no, the performance isn't great. If you've overclocked your Pi, you're using SQLite and you've set up the Cron job as described in the main text, your ownCloud installation will be fine for light usage, but you'll have to accept some sluggishness here and there. Of course, there are advantages to installing on a Pi: you end up with a silent, tiny and very power-efficient ownCloud server that you can plug into your network somewhere and then forget about.

Because current versions of Rasbian are based on Debian 7.x, you will be able to follow this tutorial without major alterations. One thing you may want to change, however, is the location of the ownCloud data directory. If you'll be using ownCloud for storing large files, it's better to move this directory off the SD card and onto an external drive. You can do this in the initial part of ownCloud configuration: when you access the web interface for the first time to create an administrator username and password, click Advanced underneath and you'll be able to assign the data directory to a different location. Just make sure that it's writable by Apache.

LINUX VOICE
TUTORIAL

ADA LOVELACE AND THE ANALYTICAL ENGINE

JULIET KEMP

Use the Linux Voice time machine to take a trip to Victorian England, and visit one of the pioneers of the computer age.

Back in the 19th century, if you wanted to do complicated mathematical calculations you had to do them by hand. To speed things up, you could buy printed tables of specific calculations such as logarithms — but as these too were calculated by hand, they were full of errors.

Enter Charles Babbage, mathematician, philosopher, engineer and inventor, who in the early 1820s designed a Difference Engine to do these calculations automatically. The Difference Engine could only add up, so it wasn't a general-purpose 'computer'. It also never existed in Babbage's time, although part of a prototype was constructed. Babbage fell out with his engineer and ran out of funding, so construction stalled around 1833 and was finally abandoned in 1842.

Meanwhile, in 1834 Babbage began to design a more complex machine called the Analytical Engine. This would be able to add, subtract, multiply, and divide, and it is the Analytical Engine that can be considered as the first general-purpose computer. Or could, if it had ever existed: Babbage built a few pieces of prototype, and carried on refining the design until his death in 1871, but never found funding for the full thing. But despite its lack of concrete existence, other mathematicians were interested in it, including Louis Menebrae, and Ada Lovelace, who was already corresponding with Babbage.

Ada Lovelace was the daughter of Lady Annabella Byron, who was deeply interested in mathematics, and Lord Byron. What would she have thought of the person who's produced Engine code that draws a cat?



Augusta Ada King, Countess of Lovelace

Lovelace had had extensive mathematical training as a child. She first met Babbage in 1833, aged 17, and corresponded with him on mathematics and logic. Around 1841 Luigi Menabrae wrote a 'Sketch' of the Analytical Engine, describing its operation and how one might use it for a calculation. Lovelace was asked to translate it into English; not only did she do that, but at Babbage's request she added her own extensive Notes, which went much further than Menabrae had.

Lovelace probably saw more in the Analytical Engine than Babbage himself had. She suggests, for example that it might act upon 'other things beside number', and that it might be possible to compose music by representing it in terms of the Engine's notation and operations. This jump from a mathematical engine to one that could act on symbols of any sort was visionary and well ahead of her time.

The Notes, importantly, contained the first computer algorithm — a series of steps of operations to solve a particular (in this case mathematical) problem. This is what any computer program does, and is what makes Ada the first computer programmer, even if she was never able to run her program on a real machine.

Installing the Analytical Engine

Although no physical Analytical Engine exists (the Science Museum in London has a working replica of the Difference engine), Fourmilab Switzerland have an emulator available. It runs on Java, so all you need to run it is a JDK. Download the emulator object code from **www.fourmilab.ch/babbage/contents.html**, unzip it, and type **java aes card.ae** from that directory to run the card file **card.ae**.

The emulator is the best guess, based on Babbage's drawings and papers over the years, of how the Engine would have worked. You can also use it as an applet, for which you'll have to download and compile the source code, but we couldn't easily get this to compile. The applet gives a more visual interface.

Basic operations and a first program

The Analytical Engine consisted of the Mill (where processing was done) and the Store (where numbers and intermediate results were held). The Store had 1,000 registers (a far bigger memory than the first 'real' computers had), and the Mill could take in two numbers, conduct an operation on them, and output a single number. The Engine would also run a printing device for output, to avoid errors in transcription. It would be operated by punch cards, as were used in Jacquard looms to weave complex patterns.

To use the emulator, then, we type in punch-card-type instructions to be run one at a time. For ease, you can put any number of cards into a single text file.

There are three types of punch cards:

- **Operation Cards** Tell the Mill to add/subtract/multiply/divide, and can also move the chain of cards forwards or backwards (like a jump or loop instruction).
- **Number Cards** Supply numbers to the Store as necessary.
- **Variable cards** Transfer values between the Mill and the Store.

For engineering reasons Babbage intended these to have three separate hoppers, but in the emulator they

go in a single stream. (This is also how Menabrea and Lovelace expressed their example programs.) The emulator 'cards' also allow some flexibility in format. Numbers aren't right-justified and there's no need for leading zeros, as there would be in a real punch card.

A number card looks like this:

```
N001 3
```

This sets column 1 in the Store (which has 0–999 columns) to the value 3.

The Mill has two Ingress Axes and an Egress Axis (plus two auxiliary axes for division, which we'll look at shortly). Once an operation is selected, the Mill will keep doing that until another is selected. The Operations cards are **+**, **-**, **x** or ***, and **/** or the divison sign, which all do what you'd expect.

Finally, the Variable Cards transfer things in and out of the Mill:

L Transfer from Store to Mill Ingress Axis, leaving Store column intact.

Z Transfer from Store to Mill Ingress Axis, zeroing Store column.

S Transfer from Mill Egress Axis to Store column.

The letter is followed by a number specifying the Store column.

A program on the Analytical Engine consists of a chain of cards; each text line in an emulator file is a single card. You submit a card chain to the Attendant, who will check it for errors and 'requests for actions' (such as inserting manually generated loops and subroutines). The chain of cards is then mounted on the Engine and processed.

Let's give it a go! Since The Analytical Engine doesn't lend itself to Hello World, we'll add 2 and 2. Save this as **card1.ae**:

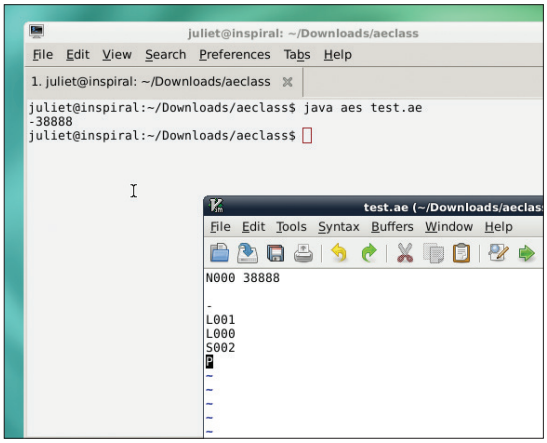
```
N000 2
N001 2
+
L000
L001
S002
P
```

This code puts 2 in column 0 of the Store, 2 in column 1 of the Store, sets the operation to add, transfers column 1 and then column 2 to the Ingress Axes (whereupon the operation will be applied), then the result back to the Store in column 2. **P** prints the result of the last operation to standard output. Run it with **java aes card1.ae** to see what happens.

In fact, you could miss out the second line, and transfer the value from Store column 0 twice, and it will automatically be transferred into both Ingress Axes. So this will work fine:

```
N000 2
+
. About to put values into Mill
L000
L000
S001
P
```

Replacing the first L000 with Z000 won't work, as this zeros the Store column after transfer. This card also includes a comment line. Comments begin with a space or a dot in column 1 of the card.



The Analytical Engine emulator running a test card (in the Vim window), which subtracts 38888 from 0.

To do more operations, you need to replace both values on the Ingress Axes – they are discarded after their use in a computation. Each time two arguments go in, the current calcuation is applied.

Menabrae and simultaneous equations

Menabrae in his Sketch described an algorithm to solve a pair of simultaneous equations. He divided the process of solving the equations into a series of individual operations, and tabulated them as Analytical Engine operations. This is handily arranged so that all the multiplications happen, then the subtractions, then the divisions, minimising the number of Operations cards.

Let's translate this into Analytical Engine code. See the LV website for the whole thing; I'll look at the structure and a couple of operations here. Here are our sample equations:

```
2x + y = 7
3x - y = 8
```

First, we put all the numbers (2, 1, 7; 3, -1, 8) into the Store. Then, following Menabrae's calculations, cards 1–6 do all the multiplying and store the results. Cards 7–9 are subtractions. Then cards 10 and 11 generate and print the results. (I've described each operation as a 'card', as Lovelace does, although in the terms of the emulator, each line is a card.)

```
Card 10 - gives x value
/
L013
L012
S015'
P
Card 11 - gives y value
L014
L012
S016'
P
```

If you're debugging, it's useful to print at every step.

Division is a little more complicated than other operations. The format is roughly the same, but dividing uses the Primed Egress Output. Specifically,

Ada Lovelace's equation for deriving the Bernoulli numbers.

$$0 = -A_0 + A_2 B_2 + A_4 B_4 + A_6 B_6 + \ldots + B_{2n}$$

$$0 = -\frac{1}{2}\frac{2n-1}{2n+1} + (\frac{2n}{2})B_2 + (\frac{2n(2n-1)(2n-2)}{2.3.4})B_4 + (\frac{2n(2n-1)(2n-3)(2n-3)(2n-4)}{2.3.4.5.6})B_6 + \ldots + B_{2n}$$

$$0 = -\frac{1}{2}\frac{2n-1}{2n+1} + (\frac{2n}{2})B_2 + (\frac{2n(2n-1)(2n-2)}{2.3.4})B_4 + (A_4\frac{(2n-3)(2n-4)}{5.6})B_6 + \ldots + B_{2n}$$

the remainder from the operation goes on the regular Egress Output, and the quotient (which is usually what you want) goes on the Primed Egress Output. You get at this by using an apostrophe. (Very large numbers can also use the Primed Ingress Axis.) Run this with **java aes simeqcard.ae** and you should get two numbers output: 3 (the x value) and 1 (the y value).

The dividing shown works fine if you have integer results or only need integer precision. But what if you want a greater precision? The Analytical Engine uses fixed point arithmetic: like a slide rule, it calculates only in whole numbers, and it is the programmer's responsibility to keep track of decimal places. So there is a "step up" and a "step down" operation, which shifts the decimal point either to the right (stepping up x times, or multiplying by 10x) or to the left (stepping down, or dividing by 10x). We just need to change the last two cards:

```
Card 10 - gives x value
/
L013
<5
L012
S015'
P
Card 11 - gives y value
L014
<5
L012
S016'
P
```

We must put the decimal point back in to the output ourselves, by manually dividing by 100,000 (105).

Ada and the Bernoulli numbers

The most interesting part of Ada Lovelace's notes on the Menabrae paper describes how to calculate the Bernoulli numbers (a set of numbers of deep interest to theoretical mathematicians) using the Engine. Her diagram of the process is too complicated to reproduce here, but can be seen (with the rest of the Notes) at **www.fourmilab.ch/babbage/sketch.html**. It can, however, be translated into code for the Analytical Engine emulator. Download the full code from the LV website; here we'll look at the structure and ideas.

The non-zero Bernoulli numbers are usually referred to by modern mathematicians as B2, B4, B6, etc. However, Ada Lovelace refers to them as B1, B3, etc. I will refer to them here by the modern numbers (so subtract one if you're comparing with the Notes directly). There are many ways to derive them, but the equation that Lovelace uses is shown, left. Note that the very last Bernoulli number has no accompanying A-equation. What we're trying to calculate.

The important point is that from A2 onwards, each following A-value takes the preceding one and multiplies by another two terms. This makes it possible to construct an iterative process to calculate each succeeding term.

Onwards then to the code! Following Lovelace's diagram, we will put in an already-calculated version of B2, B4, and B6, and will calculate B8, so **n** is 4. As Lovelace was keen to point out, in a 'real' calcuation the Engine itself would have already calculated these values on a previous round of the program, so they're stored in a later register. The first section of the code, then, sets up our numbers. Register 3 holds our **n**, and registers 21–23 the first 3 Bernoulli numbers, multiplied by 10,000 (to allow for later dividing, as discussed above).

Cards 1-6 calculate -1/2 x (2n - 1)/(2n + 1). The last three are the most interesting:

```
Card 4: (2n - 1) / (2n + 1)
/
L004
<5
L005
S011'
Card 5: 1/2 * (2n - 1) / (2n + 1) Y
L011
L002
S011'
Card 6: -1/2 * (2n - 1) / (2n + 1) Y
-
L013
L011
S013
```

In Card 4, we step the first value up 5 places before dividing, to avoid a rounding error. In Card 5, we take the value stored in the previous step and overwrite it, since it won't be needed again. In Card 6, we take advantage of the fact that any unused register reads 0, to get a minus number by subtracting register 11 from zero. Effectively this switches the sign of the value in step 5, but we store this result in register 13.

Card 7 subtracts one from n. This isn't used in the code as it stands, but it is a notional counter to keep track of whether we need to do another round of calcuation. If we were calculating B2 (so n = 1), then card 7 would give the result 0, and we would be done. Otherwise, it should add 1 to n and go round again. Lovelace presupposed that the Analytical Engine would have a way of detecting a specific result and acting accordingly. (The emulator provides an alternation card to do exactly this.)

Steps 8–10 produce (2n / 2) * B2 (the latter being stored already). Card 11 adds the value from the first

stage (A0), and card 12 again checks whether we're finished yet.

The intriguing part is the next stage, cards 13–23. This is the section that could be repeated almost exactly for any stage of the process, however many numbers you wanted to calculate. What you need to calculate each time is:

```
2n . (2n - 1) . (2n - 2) ... / 2 . 3 . 4 ...
```

This is equivalent to

```
2n / 2 . (2n - 1)/3 . (2n - 2)/4 ...
```

The first time we go through the loop, when calculating A3, we can forget about 2n / 2 as we already calculated that on card 9, and saved it in location 011. So we work out 2n - 1 (card 13) and 2 + 1 (card 14), divide them and save the result (card 15; note again that we step up 5 decimal places), and then multiply it with A0 and save this new value in location 11. We then repeat the exercise, with cards 17-20, with (2n - 2) / 4, multiply it with the previous result, and overwrite location 011 again. So, once again, our A-value is stored in location 11.

In card 21, we multiply with our pre-saved value for B4, then add the whole sequence up and save it in location 13. Card 23 once again checks for 0.

At this point, all we need to do is to run cards 13–23 all over again. Because we saved 2n - 2 as our 'new' 2n, in location 6, applying cards 13–16 produces the result (2n - 4)/ 5, just as we want. And the same again for cards 17-20, with (2n - 5) / 6 multiplied in this time. The only change is that in card 21, we have to grab B6 from its location rather than B4. Then we add it all together again. In the code, these second-time-around cards are labelled 13B-23B.

```
 Card 13: 2n - 1 Y
L006
L001
S006
 Card 14: 2 + 1 Y
+
L002
L001
S007
 Card 15: (2n - 1) / (2 + 1)
/
L006
<5
L007
S008'
 Card 16: (2n / 2) * ((2n - 1) / 3) Y
*
L011
L008
S011
 Card 17: 2n - 2 Y
-
L006
L001
S006
 Card 18: 3 + 1 Y
+
L001
L007
S007
 Card 19: (2n - 2) / 4  Y
/
L006
<5
L007
S009'
 Card 20: (2n / 2) * (2n - 1)/3 * (2n - 2)/4  Y
*
L009
L011
>5
S011
 Card 21: B(4) * [Card 20]
L022
L011
>5
S012
 Card 22: A0 + B2A2 + B4A4 Y
+
L012
L013
S013
```

There's only one new thing to notice, which is that in cards 20 and 21 we have to step our result from the multiplication back down by five decimal places, as we're multiplying two stepped-up values together.

The final step is 24, in which we add our saved value from step 23 to a zero register, to give our calculated Bernoulli number. In actual fact, we should be subtracting this from zero to get the sign of the number correct, but Lovelace explicitly chose to ignore this. Once the result is output, remember that you'll also need to manually put in the decimal point, five places to the left. So our result is -0.03341.

This is not far off the 'official' -0.033333333. Try altering the accuracy of our calculations (remember also to alter the accuracy of the stored Bernoulli numbers) to improve the accuracy of the result.

The Analytical Engine emulator also supports looping code, using conditional and unconditional cycle (backing) cards, and straightforward backing/advancing cards; and an if/then clause with the alternation card. See the website for more details, and have a go at rewriting the provided code to loop over one Bernoulli number at a time, up to a given n, generating the result and storing it for the next loop around. Remember that you'll need to calculate A0, A2, and B2 separately, as here (cards 1–12), before you can get into the real 'loop' part. As the emulator is Turing-complete you can also, as Lovelace suggested, produce anything you can translate into Engine-operations; or, as we now think of it, assembly language. In theory you could even write a compiler in Engine code...

Juliet Kemp is a scary polymath, and is the author of O'Reilly's *Linux System Administration Recipes*.

LINUX VOICE TUTORIAL

ARCH LINUX: BUILD A POWERFUL, FLEXIBLE SYSTEM

GRAHAM MORRISON

Install the rolling release distro of the moment and you'll never have to wait for a package upgrade again.



DIFFICULTY

Installing Arch is the Linux equivalent of base jumping. You organise yourself. Surround yourself with everything you need, stick the installation media on to a USB stick and jump. You never know how an installation is going to go until you try it, and it will always involve a bit of *ad-hoc* hacking, Googling and troubleshooting. But that's the fun of it, and that's what makes Arch different.

With Arch, you're on your own. In a world where where technology is taking your personal responsibility and giving it to the cloud, or to an internet search filter or the device manufacturers, getting your hands dirty with an operating system can be a revelation. Not only will you learn a great deal about how Linux works and what holds the whole thing together, you'll get a system you understand from the inside-out, and one that can be instantly upgraded to all the latest packages. You may also learn something about yourself in the process. And despite its reputation, it's not that difficult.

If you're a complete beginner, you may need to hold on to your hat, because installing Arch is an uncompromising adventure in core tools and functions. It's a jump into the unknown.

1 CREATE THE INSTALL MEDIA

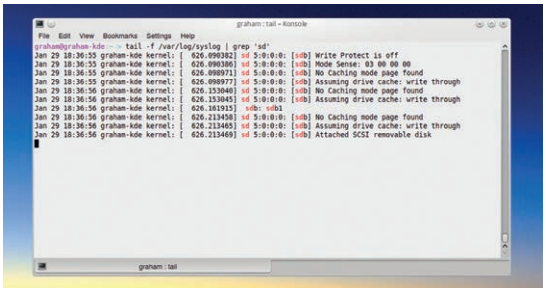
We'll start with the ISO, which you can either find on our cover DVD or download from your local Arch mirror (see **https://www.archlinux.org/download**). If you're going to install Arch onto a machine with a DVD/CD drive, you could simply burn the ISO to a blank CD, but we're going to write the ISO file to a USB thumb drive as this saves wasting a disc. You'll only need a 1GB thumb drive but this process will remove all data from the device, so make sure there's nothing on there you want to keep first.

There are many ways of transferring an ISO image to a USB drive, although copying the ISO onto the filesystem isn't one of them. Normally, our preferred method is to use the graphical tool UnetBootin, which is available for nearly all distributions, including those two alien environments, OS X and Windows. Sadly, Unetbootin won't work with Arch unless you manually edit the **syslinux.cfg** file afterwards, as this is overwritten in the transfer process. This leaves you to the mercy of **dd**, a crude command that copies the raw data from one device to another. It works, but there's no sanity checking of the output device you choose, so you have to make sure you're writing to your USB stick. If you get this wrong, you'll copy the raw bits and bytes of the Arch ISO to another storage device on your system, overwriting any data that might have been there before.

Here's our system for getting the correct device:

1. **sudo tail -f /var/log/syslog | grep sad**
2. **Clear your terminal window buffer**
3. **Plug in your USB drive and watch the output**

You'll see several lines appear as your system negotiates with the new USB device and, all output will include the characters 'sd'. What you need to look for is the letter that comes after 'sd', as this is the device node of the USB stick after it's connected to your system, and we need this device name for the next command, which is going to write the Arch ISO image to the USB stick. Also be aware that this device node can change, if you come back to this process after adding or removing another USB device. Here's the **dd** command for writing the ISO:



Whenever a new USB device is connected, your system logs become a hive of activity

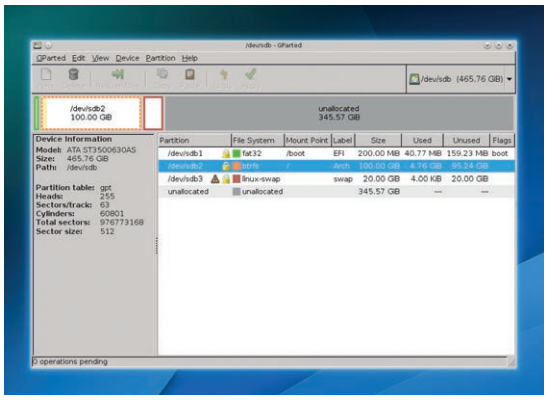
```
sudo dd bs=4M if=/path/to/arch.iso of=/dev/sdx
```

Replace the **x** in **sdx** with the letter for your device and press return. You should see the activity LED on your USB stick start to flicker as data is written. If not, press Ctrl+C immediately to stop the process and double-check everything (such as whether your USB stick has an activity LED). After the process has completed, which should only take a few moments on a modern machine, type **sync** to make sure the write buffers are flushed, and remove the stick. It's now ready to be used to install Arch.

2 FIRST BOOT

Before you plug the USB stick into the machine on which you're going to install Arch, make sure you know which hard drive you're going to use. If your machine has several drives, make a note of the capacity and model of the drive you want to use, and make sure you don't have an identical drive. If you're going to use a partition on a drive, or use up free space, we'd recommend using GParted from a live CD to set up your partitions first, or at least resize other partitions to leave enough space.

Along with a 200MB EFI partition for GUID, you'll need at least a root partition and a small swap partition. It may also help to have a separate home partition, as this makes upgrades to the root filesystem easier to handle. Most machines will boot off the USB drive by selecting the custom boot menu from your machine's boot flash screen. It's usually done by pressing the F12 key. This will present you with a list of connected drives, and you should be able to select the USB device from there. If all goes well, a moment later you'll see the Arch boot menu and you need to select the first option, 'Arch Linux archiso'.



We used GParted to create a GPT partition scheme and a 200MB EFI partition (type ef00, labelled 'EFI"). But it might be easier to stick with old-school MBR and Grub.

Networking

Your first mission is to get to the internet. We'd recommend installing the system using a wired connection if at all possible. With the system up and running, it's then much easier to configure your wireless device, but if you need to configure wireless now, check out the excellent Arch Beginners' Guide.

With a bit of luck wired internet should be working already, because Arch runs the **dhcpd** daemon at startup, which in turn attempts to get an IP address from whatever router your kernel-configured network interface can find. Try typing **ping linuxvoice.com** to see if any packets are returned. If this doesn't work – and it didn't for us – first get the name of your interface by typing **ip link**. It's usually the second device listed, because you should ignore the first one called **lo** (this is a system loopback device). Our PC's network device is called **enp7s0**, which you'll need to replace in the commands below. To get it working, we stop the non-functioning DHCP service, bring up the Ethernet interface, manually assign this to a valid IP address on our network and add the router as a default gateway. If you know your router's IP address, you can normally connect to its web interface to check which IP ranges are suitable for your machine, and use its IP address as the router IP address. Here are the three commands to do what we just explained – replace IP addresses to suit your own network.

```
ip link set enp7s0 up
ip addr add 192.168.1.2/24 dev enp7s0
ip route add default via 192.168.1.1
```

The final step is to type **nano /etc/resolv.conf** and add the line **nameserver 8.8.8.8** to add one of Google's nameservers to the mix. This will convert the alphanumeric URLs we normally use to the IP addressees used by the network, and you should now find that pinging a domain name works.

LV PRO TIP

In this tutorial we've chosen EFI booting and the GUID partitioning scheme, as this is likely to be compatible with most hardware available now, and more future proof than MBR partitioning.

3 FORMATTING

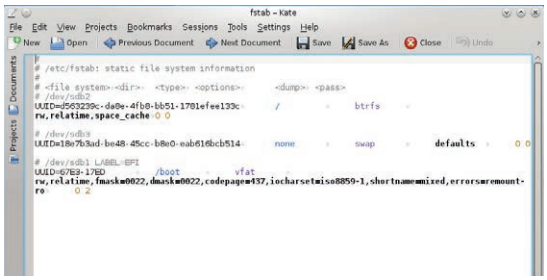
You should now have a fair idea at how Arch does things. It basically leaves you to do your own research and make your own decisions while creating the most common-sense environment it can. We're going to assume you've already partitioned the drive, so the first step is to make sure you know which drive to target. The best command to achieve this is **fdisk -l**. This lists all your drives, their partitions and the filesystems they're using, alongside their device nodes. Unless you've got two identical drives, you should be able to work out which one to use without too much difficulty. And if you haven't formatted your new partitions yet, they should stick out like a sore thumb. If you're only using a single drive, you'll have even fewer problems. We do know people who disconnect all other drives whilst installing Linux so that they can be absolutely sure they won't get the wrong drive and overwrite their 500-hour *Skyrim* save position on Windows 7.

Choose your filesystem

You should now format the partition. The safest and most sensible filesystem to use is ext4, and you can format your chosen partition by typing **mkfs.ext4 /dev/sdx2** – again, replace **x2** with your own partition. You should do this for your home partition too, and you will also want to format and define your swap partition. The command to do this is **mkswap /dev/sdx3**. You can turn this on with **swapon** followed by the device node. If you created an EFI partition yourself, rather than another OS doing this, you can format it with the command **mkfs.fat -F32 /dev/sdx**.

LV PRO TIP

Arch's own docs are absolutely excellent. They're also very comprehensive, so don't allow them to put you off.



Our automatically generated fstab file didn't need any further edits

Now mount the partitions by typing:

```
mount /dev/sdx2/ /mnt
mount /dev/sdx3 /mnt/home
```

With GUID and an EFI system (rather than using the old BIOS), you'll also need to mount the EFI partition:

```
mount /dev/sdx1 /mnt/boot
```

If you're not using a separate home partition, type **mkdir /mnt/home** to create a home folder in the root partition. These are the fragile beginnings of your Arch installation. We're going to make more of an impact with the next command:

```
pacstrap -i /mnt base
```

This command installs a basic Arch system to your drive. We leave the installer at its default settings so it can grab and install all the default packages, and you'll be left with all the packages you need. However, unlike with other distributions, that doesn't mean it's actually usable for anything yet. Following the Arch Beginners' Guide, we'll next create the **fstab** file, as this tells the distribution where to find its dependent filesystems. In the old days, we'd use labels to represent partitions, but labels can be changed or duplicated and break an **fstab** file, so we now use UUIDs. These are basically hashes derived from partition data, so Arch should never get confused unless something changes with the partition scheme. The correct file with the correct mount points and UUIDs can be generated automatically by typing:

```
genfstab -U -p /mnt >> /mnt/etc/fstab
```

You can see that this file is created in your new root filesystem, and as the file was generated automatically, you should check it's not complete insanity (try **cat /mnt/etc/fstab**). It will show your mounted filesystem along with the EFI partition we mounted on **/boot** – this should be formatted and listed as **vfat**, as per our formatting command earlier. With all that set up, we're now going to teleport ourselves into the new Arch system using 'chroot' with the following command:

```
arch-chroot /mnt /usr/bin/bash
```

4 POST-CONFIG

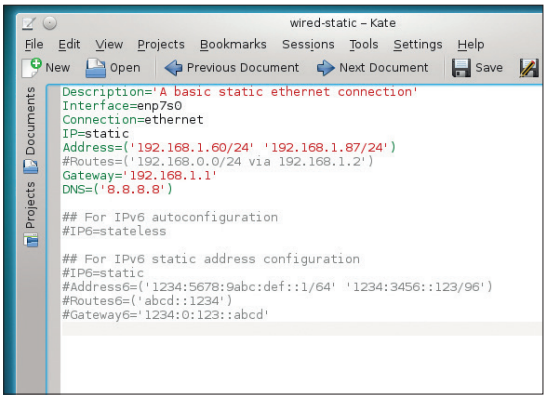
How does it look inside your new Arch installation? Not that different than from the USB stick, except for now you're executing code from your hard drive. There's obviously lots we can do here, but we're mostly interested in getting the system up and running as quickly as possible. It's worth giving your machine a hostname, which can be done with a command like **echo linuxvoice > /etc/hostname**. Networking too should be solved in exactly the same way we got networking working earlier. If DHCP worked, just type **systemctl enable dhcpcd.service** to make the required link to get it running at boot.

LV PRO TIP

Despite updates being easy to apply on the command line, it's always worth checking that nothing requires your intervention before you do the upgrade. The best way we've found to stay in touch is to peruse Arch's Twitter account: **@archlinux.**

Enable network profiles

An alternative to this generic solution, which didn't work for us, is to enable network profiles, such as the ones mainstream distributions use to quickly switch between network settings. First copy the **/etc/netctl/examples/ethernet-dhcp** file to **/etc/netctl/** directory, open your new file with Nano and change the device from **eth0** to whatever your machine uses (take a look at the output from **ip link**), then enable the connection for your next boot with **netctl enable ethernet-dhcp**. If you want to do the same with a static IP address, use the static Ethernet example configuration. But for this, you have to make sure DHCP isn't running when the system starts. To remove it, and any other service you no longer require, the command is **systemctl disable dhcpcd.service**. Arch now uses systemd, which is why this syntax may look unfamiliar. You can check the service isn't started automatically by typing **systemct | grep dhcp** when you next boot. If you want netctl to automatically bring up a connection for your interface, whether you've configured it for a static or dynamic connection, type the following, but replace **enp7s0** with the name of your device:



We had to create a static networking configuration file and remove the DHCP service to get networking working.

```
systemctl enable netctl-auto@enp7s0.service
```

Before leaving the chroot environment, set a password by typing **passwd**, then **exit** and **reboot**.

We've now got to the state where we've got enough installed and configured that we can finally breathe some native life into our distribution. But before we can reboot, we need to install a bootloader. If you've already got Linux installed, or you're sharing an installation with Windows, you'll need to be careful. Installing a bootloader over a part of the disk used by

another operating system will stop that other operating system from booting. If you've dedicated a new single drive to Arch, which is what we'd recommend, you can install the bootloader onto this drive only – whether that's old-school MBR or newer GUID. This way, you won't break anything; your drive will boot if it's the first boot device, and it will boot if you use your system's BIOS boot menu and select an alternative drive. If you want to add your Arch installation to another Grub installation, you'll need to boot into that system and re-generate the configuration – many distributions, such as Ubuntu, can do this with a minimal of effort.

Install a bootloader

As we're using a modern system with EFI and GUID partitioning, we're going to install a simple EFI bootloader rather than the more commonly used Grub. If you are using older partition, however, Grub can be installed with the following two command after changing **sdx** to your device:

```
pacman -S grub
grub-install --target=i386-pc --recheck /dev/sdx
```

For EFI systems, type **pacman -S gummiboot** to install the EFI bootloader package, and **gummiboot install** to run the simple setup procedure. It will fail if an EFI-compatible partition can't be found, or isn't mounted. If that happens, you should install Grub.

The only other step to getting **gummiboot** to work is to create a simple configuration file called **/boot/loader/entries/arch.conf**. It should contain the following information:

```
title     Arch Linux
linux     /vmlinuz-linux
initrd    /initramfs-linux.img
options   root=/dev/sda2 rw
```

Replace the **sda2** part with the device node for your root partition and your new system should work. If it doesn't (and we don't want to be negative, but this is Arch we're talking about), the great thing about the Arch USB installer is that you can easily use it to troubleshoot your installation using the skills you've already learnt. Just reboot from the USB stick, mount the drive and **chroot** into your new Arch installation. Many serious problems can be solved this way, and it's much quicker than using a live CD. Remember this as you type **exit** to quit the **chroot** environment and **reboot** to restart your machine, because if your new Arch installation doesn't appear, you'll need to boot again from the USB stick and check the configuration,

LV PRO TIP

Pacman is Arch's package manager, and is relatively straightforward to use. **-S** will search for and install packages; **-Ss** will search for package names and their descriptions; **-R** will remove them and **-Syu** will perform a system upgrade.

5 BUILD YOUR OWN HOME

You now need to log in as root, and you should check that networking is working. If not, you need to go through the same steps we went through with the USB installer.

At its most basic level, Arch is now installed and ready for you to sculpt into your perfect distribution. There are many ways to do this – you may even want to remain on the command line, but we're going to assume you'll want a graphical environment and your hardware working. Xorg, the graphical display server, can be installed with the following command:

```
pacman -S xorg-server xorg-server-utils xorg-xinit xterm mesa
```

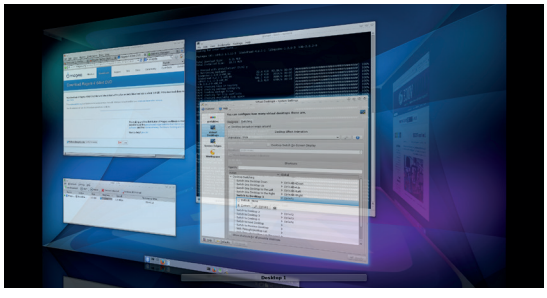
As long as you're happy using open source drivers for your graphics hardware, this is all you need for a working X session. Many of the open source drivers are good enough for desktop work, and only lack 3D performance. A simple test to make sure all this auto configuration is going to work is to type **startx** to bring up the most basic of X sessions. Unfortunately for us, it didn't work and we got a 'no screens found' error. This is probably because our screen is rubbish and isn't communicating its capabilities back to the graphics hardware. The solution is to create your own X.org config file. We're using Nvidia hardware and are happy to use Nvidia's proprietary drivers. The drivers for any modern Nvidia GPU can be installed by simply typing **pacman -S nvidia**, and rebooting your system. Nvidia's drivers are also better at detecting displays, so it might be worth trying **startx** again to see if anything has changed. You can quit the X environment by exiting all of the terminal sessions.

With X running, it's now time to install a graphical environment. Obviously this is a contentious issue, but here's the basic procedure. KDE, for example, can be installed by typing:

```
pacman -S kde-meta
```

Meta packages encapsulate other package collections, so you can fine-tune your installation. A basic KDE installation can be accomplished by grabbing the **kde-base** package, for example. **kde-meta** on the other hand downloads over 700MB of data and installs over 2GB from 558 packages. It takes a while. For Gnome, **gnome-shell** contains the basics, **gnome** has the desktop environment and the applications, while **gnome-extra** contains all the tools.

The final steps to Arch nirvana are to create a new user with **useradd -m graham**, give them a password with **passwd graham** and then to launch the KDE/Gnome login manager by typing **kdm** or **gdm**. You'll get a fully functional login and desktop. But as you'll soon discover, this is only the end of the very beginning. With Arch, you've only just got started.



This being Arch, you don't have to install KDE. But when was the last time you saw a gratuitous screenshot of the desktop cube looking so good?

LINUX VOICE TUTORIAL

PYTHON: BUILD DYNAMIC WEB PAGES

BEN EVERARD

Keep your websites up to date by harnessing live data piped from Python's web services.

WHY DO THIS?

- Keep your websites up to date with the latest information
- Pull data from across the web and feed it into your programs
- Discover the powerful combination of Python and Tornado

HTML is one of the greatest developments in computing. It's so simple, anyone with a text editor can knock up a simple web page with text, pictures, and links to other sites. This simplicity gives the web the potential to grow to encompass almost the whole of humanity. However, its original developers intended it for content that doesn't change very often. Every change to a page of HTML needs the file to be modified and resaved, which is fine for some purposes; but sometimes you need something a little more dynamic. In this tutorial we're going to look at four different methods for including constantly changing content in your website.

Since we've got a lovely new magazine, we're going to create a lovely new website to help us keep track of everything that's going on. The skeleton code for this website is:

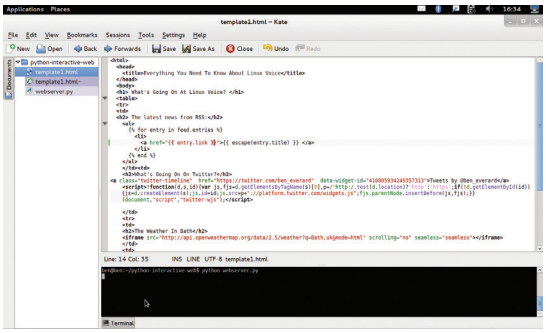
```
<html>
<head>
<title>All About Linux Voice</title>
</head>
<body>
<h1>All About Linux Voice</h1>
<table>
```

Useful Tornado template

Tornado templates are based on Python, but they have their own simple language. Here are a few of the most useful commands:

- **{% set var_x = a_value %}** Sets local variable **var_x** to the value **a_value**.
- **{% if condition_1 %} … {% elif condition_2 %} … {% else %} … {% end %}** An if statement. elif and else are optional.
- **{% while condition_1 %} … {% end %}** A normal while loop.
- **{% import a_module %}** Import the Python module **a_module**.
- **{% include a_template %}** Copy the contents of **a_template** to this point in the file.

There are full details of the template syntax at **www.tornadoweb.org/en/stable/template.html**. In general, it's best to do as much of the processing as possible in the web server, and use the template just to display the data. You can use the included commands to create various components that you can combine in different ways on different pages.



A good text editor will highlight different parts of the code, so you can see what part does what.

```
<tr>
<td>Data1</td>
<td>Data2</td>
</tr>
<tr>
<td>Data3</td>
<td>Data4</td>
</tr>
</table>
```

If you haven't come across HTML before, everything is kept between pairs of angular bracketed tags that describe what the content is. For example, **<h1>** marks the start of heading 1 (the biggest heading), and **</h1>** tells the browser that we've finished the heading. The **<table></table>** tags describe a table, **<tr></tr>** describe a table row, and **<td></td>** describe a table cell. The skeleton code can be found on the coverdisc or at **linuxvoice.com** as **lv-skeleton.html**.

In this skeleton, Data1 to 4 are the places we'll put four different pieces of dynamic content.

As a British magazine, the most important thing to us is obviously the weather, and this changes a lot. If we kept looking out of the window, and updating our website every time the weather changed, we'd have no time to make tea, let alone a magazine. Fortunately, though, we don't have to. The first, and easiest, method of including dynamic content we'll look at is an iframe. These enable you to embed other websites inside yours. In this case, we'll embed a weather forecast. You can put in any website, though it's best to do it with one designed for the purpose, otherwise it's unlikely to look good. For our purposes, **openweathermap.com** provides exactly what we need. The website **http://api.openweathermap.org/data/2.5/weather?q=Bath,uk&mode=html** is a

compact forecast for the beautiful city of Bath, designed for embedding.

In the skeleton, you can change **Data1** to the following:

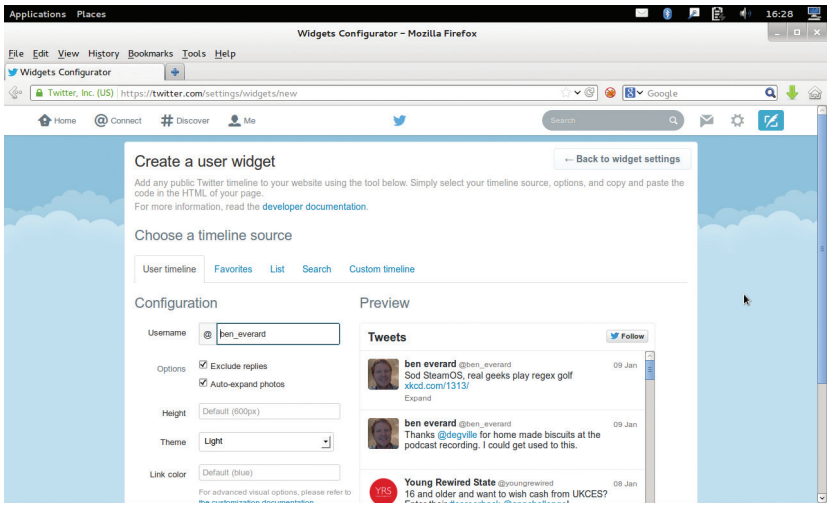
```
<h2>The Weather In Bath</h2>
<iframe src="http://api.openweathermap.org/data/2.5/weather?q=Bath,uk&mode=html" scrolling="no" seamless="seamless"></iframe>
```

This will embed the weather forecast in our website. The **scrolling** value tells the browser that we don't want a scroll bar on the iframe, and **seamless** tells it that it should be integrated into our page seamlessly. Not all browsers recognise these, so it will appear slightly different on different platforms.

Keepin' Tweetin'

Iframes are the most basic way to grab data and serve it in a web page. For simple things like weather forecasts they work great, but sometimes they're a bit lacking. Some data providers provide 'widgets' that you can put in your page. These are generally small chunks of HTML, usually with some JavaScript to grab data and display it in a useful way. For our Linux Voice website, we'll add a widget that grabs the Linux Voice Twitter feed.

You can create Twitter widgets for any Twitter account. On the Twitter web page, just go to the cog menu icon, then Settings > Widgets > Create New. By default it'll set it to the currently logged-in account, but you can change this to whatever you like. We also changed ours to have a height of 300. Once you've entered the details and hit Create, the appropriate code will be displayed below the preview. You just need to copy and paste it in place of **Data2**. The code



You can create Twitter widgets to show anyone's tweets, but if they post something inappropriate, it will be displayed on your site as well.

we used was:

```
<a class="twitter-timeline" href="https://twitter.com/LinuxVoice" data-widget-id="419158898222698496">Tweets by @LinuxVoice</a><script>!function(d,s,id){var js,fjs=d.getElementsByTagName(s)[0],p=/^http:/.test(d.location)?'http':'https';if(!d.getElementById(id)){js=d.createElement(s);js.id=id;js.src=p+"://platform.twitter.com/widgets.js";fjs.parentNode.insertBefore(js,fjs);}}(document,"script","twitter-wjs");</script>
```

Of course, you can create your own.

Don't worry about trying to understand this script (unless you're a JavaScript masochist) as it's computer generated and not meant to be human readable. Save the file and refresh your web browser and you should now have the weather and the latest news from Twitter all without having to handle any of it yourself. There are a few options on the Create Widget Twitter page to help you control the look and feel of this datastream, so see which settings work best with your page.

Get more control

The problem with the two previous methods is that they pull everything from the other website, so as well as the data you get the other site's formatting too. Sometimes this isn't a problem and the simplicity is worth it. Other times you may find that you want more control over how the content is displayed, or even the ability to process it before putting it on the screen. Another risk in putting content from other places on your website is that they could maliciously alter your page using JavaScript. It's unlikely that either Twitter or OpenWeatherMap would do this deliberately, but if hackers managed to break into the main system, they could use this to attack all the web pages that pull data from there.

"You may find that you want more control over how the content is displayed"

Therefore, it's better if you don't just put other people's content directly into your site, but process the data and produce HTML that uses the raw data. For this we're going to use Python.

Make it more dynamic

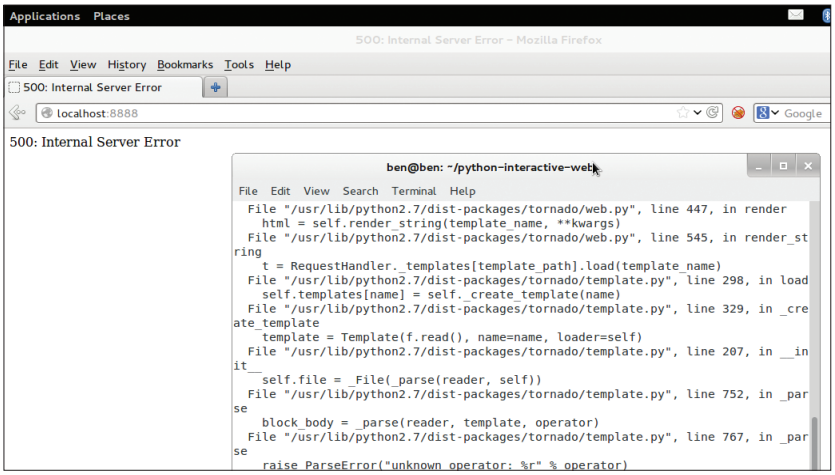
Server-side processing is great for keeping a site updated, but it has one fatal flaw: it only updates the information every time the website is loaded. Sometimes you need to keep a page's information fresh even if the user leaves it loaded.

The simplest solution is simply to tell the browser to keep refreshing the page. This is incredibly simple – just add the following tag inside the **<head></head>** tags of the template:

```
<meta http-equiv="refresh" content="60" >
```

The **content** value is the number of seconds after loading you want it to refresh.

This method is a little crude, but it will work. A more advanced technique is to keep a connection open between the browser and the server and continue to send data between them. There are ways of doing this using HTTP, but a better solution is to use websockets. These require both code on the server and JavaScript running in the browser in order to work properly, and they're a bit beyond the scope of this tutorial, but you can find out how to use them on the Tornado website at **www.tornadoweb.org/en/stable/websocket.html**.



If there's a problem with the template, the site won't load. You'll get Python errors, but they aren't usually very helpful.

The Tornado module contains a web browser that lets you modify templates by passing more information to them. To start with, you'll need to make sure you have the appropriate Python modules installed. We'll be using Tornado and Feedparser (as well as some modules from the Python standard library). These are available through the PIP (Python Install Python) package manager for Python, but it'll be easier to keep them up to date if you install them through your distro's package manager. On Debian-based systems you can do this with:

```
sudo apt-get install python-tornado python-feedparser
```

Once this is done, you just need a simple Python program to serve the website. We've called this code **webserver-start.py** and it's on the DVD and website.

```
import tornado.ioloop
import tornado.web
class MainHandler(tornado.web.RequestHandler):
  def get(self):
    self.render("lv.html")
application = tornado.web.Application([
  (r"/", MainHandler),])
if __name__ == "__main__":
        application.listen(8888)
        tornado.ioloop.IOLoop.instance().start()
```

We won't go into everything that's going on here (you can learn more about Tornado from the excellent documentation at **www.tornadoweb.org/en/stable**), but simply put, this starts a web server on port 8888 of localhost. It has a class called **MainHandler**, which is used every time someone visits the root of this web server (ie **r"/"** in the above code). The method get of this class is called every time someone sends a GET HTTP request to this address, and it renders the template **lv.html**. (Make sure the HTML file you created before is called **lv.html**). As long as you save this in the same directory as **lv.html**, you can run it from a terminal in the same directory with:

```
python webserver-start.py
```

Once that's running, you can point a web browser to **http://localhost:8888** and it'll display the same page as before. The difference is that it's now a Tornado template, which has more power than regular HTML, and you can pass it data from the Tornado server.

A Yen a Mark a Buck or a Pound

As Linux Voice does a lot of business in the USA, changes in the exchange rate between the Dollar and the Pound make a difference to our income. Keeping tabs on this is important, so the next bit of data we pull in will be the latest exchange rate.

www.openexchangerates.org operates a service that enables you to grab the latest exchange rate data (you'll need to register for an API key before you can use it though). There are various levels, but the free one is suitable for our needs, and you can sign up for it here: **https://openexchangerates.org/signup/free**.

The data comes in JSON (JavaScript Object Notation) format. While this was designed for JavaScript, it also works really well with Python.

There are a couple of Python modules that help us get and access the data: **urllib2** and **json**. The code to grab and access the data is:

```
import urllib2
import json
def getRate() :
        url = "https://openexchangerates.org/api/latest.json?app_id=YOUR_API_KEY"
        req = urllib2.Request(url)
        response=urllib2.urlopen(req)
        return json.loads(response.read().decode("UTF-8"))['rates']['GBP']
```

This piece of code needs to go into the **webserver-start.py** file between **import tornado.web** and **class MainHandler**. Change **YOUR_API_KEY** to the one you got when you signed up for the service.

urllib grabs and opens the resource, then the **json** module converts it into a Python dictionary. This has the key rates, which is another dictionary, and the key GBP returns the Dollars–Pounds exchange rate. You then need to pass the latest data across to the template by changing the line **self.render("lv.html")** to:

```
self.render("lv.html", rate = getRate())
```

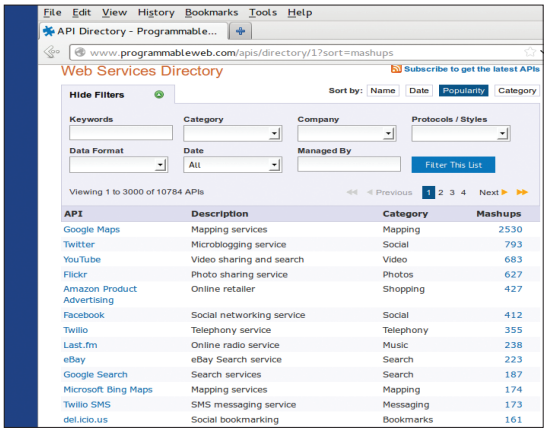
Data sources

There are loads of places you can get information for dynamic websites. OpenWeatherMaps provide JSON-encoded weather data for forecasts as well as current weather. Twitter also has an API that's easy to use through a module such as **python-twitter** (**http://code.google.com/p/python-twitter**).

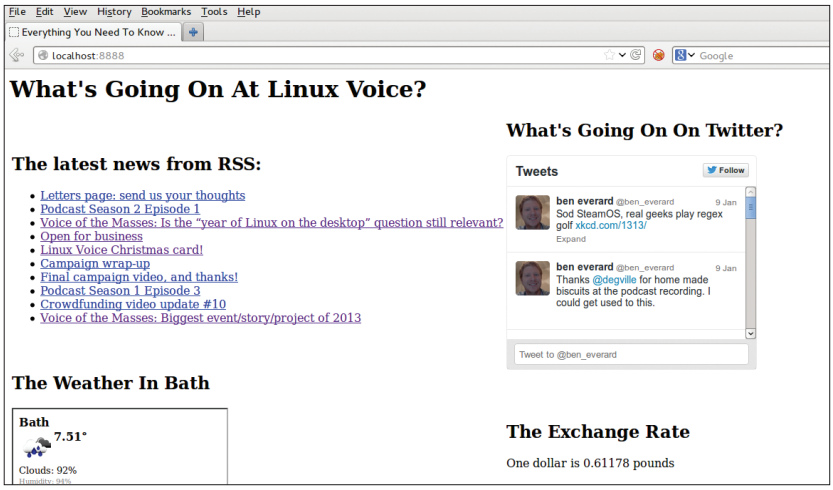
In addition to the ones we've looked at here, these are some more that you may find useful:

- **Facebook Graphs API** (**https://developers.facebook.com/docs/graph-api**)
- **IPInfoDB** (**http://ipinfodb.com/ip_location_api_json.php**) enables you to check the location of an IP address.
- **The BBC** (among others) publishes an RSS feed of the latest news. It also has a few APIs to help you access information about what's happening.
- **Reddit** can be browsed through JSON. For an example, take a look at **www.reddit.com/r/linux/hot.json**. For more information see **www.reddit.com/dev/api**.
- **StopForumSpam** hosts a database of known spammer IPs that you can use to vet visitors, though there are some restrictions on use. Take a look at **www.stopforumspam.com/usage**

These are just a few examples; there is a huge range of data sources available. Many offer free access, but some are only for paying customers.



You'll find a comprehensive list of useful data sources at www.programmableweb.com.



We focused on simplicity so it's not pretty to look at, but there are plenty of CSS and HTML tricks to sort that out.

This created the global variable rate that you can access in the HTML template with **{{ rate }}**. Change **Data3** to:

```
<h2>The Exchange Rate</h2>
One dollar is {{rate}} pounds
```

After you make any changes to either the webserver code or the template, they won't take effect until you restart the web server (a simple Ctrl+C to stop it, then re-running **python websever.py** does this). You can then refresh the website in the browser. If everything's worked correctly, you should see the exchange rate displayed. You can use this method to put whatever you want into the web page. This could be things you've just pulled from a database, or information about the computer that you're running on as well as data grabbed from other sources.

Going Loopy

Tornado templates can do far more than just display the values that are passed to them. They can also include bits of Python code that can manipulate the data. The final piece of our datastream will demonstrate this. We'll pull in the latest posts from the Linux Voice website using RSS and the **feedparser** module. This works a bit like the **json** module in that it pulls in data and converts it into a Python dictionary. However, unlike in the previous example, this time we'll pass the entire dictionary to the template and process it there. You'll need to add the line

```
import feedparser
```

To the start of **webserver.py**, then change the get method of MainHandler to:

```
lv_feed = feedparser.parse('http://www.linuxvoice.com/feed/')
self.render("lv.html", feed = lv_feed, rate = getRate())
```

You don't need to use **urllib2** to get the document with RSS, as the feedparser module handles everything.

On our page, we want to loop through every entry in the RSS file and display the post title as a link to the post on **linuxvoice.com**. In the template, you can include Python code inside **{% %}** brackets. Indentation doesn't work; instead, blocks of code are ended using **{% end %}.** This is done with the following code (in place of **Data4**):

```
<h2> The latest news from RSS:</h2>
<ul>
{% for entry in feed.entries %}
<li>
<a href="{{entry.link}}">{{ escape(entry.title) }} </a>
</li>
{% end %}
</ul>
```

The final version of **webserver.py** is on the website and DVD as **webserver-final.py**.

**
** creates an unordered (ie bullet pointed) list, and **
** tag items in the list.

This for loop repeats every line between it and the end, including the HTML lines. This will then create a new list item for every item in **feed.entries**. The **escape()** function just adds escape characters to the text before passing them across so they display correctly in the browser.

As you can see, any data that you can access with Python, you can display on a website. Tornado templates give you complete control over how these are displayed. If you're already running a website with Apache, it isn't easy to incorporate this last technique into it, though you could do something similar with PHP or even JavaScript. If you're using Nginx as a web server, you could set it up to reverse-proxy a Tornado server for some pages while retaining the speed of Nginx for simpler pages.

Ben Everard is the co-author of *Learning Python With Raspberry Pi* – coming soon to an Amazon near you.

XML

We've looked at JSON, HTML and RSS for data sources, but they're not the only options. XML is also a common format for data on the web, though it can be a little more complex than the others. It's often done using ElementTree. As the name suggests, this converts the XML into a tree from which you can then extract the information you need.

LINUX VOICE TUTORIAL

EUCLID'S ALGORITHM: RECURSION AND PYTHON

GRAHAM MORRISON

Learn a wonderfully simple algorithm that teaches as much about Python as it does about mathematics.

WHY DO THIS?

- You'll learn ancient wisdom about numbers and their factors.
- This is a great way to see how Python deals with Boolean operations.
- While at the same time, see the danger of putting everything into one line.

We're about to go back to the year 300 BC. A time when much of the world looked like the cover of the Led Zepplin album *Houses of the Holy*. This is the time of Euclid; mathematician, Greek geek and founder of all things geometrical.

The problem that Euclid's algorithm solves is easy enough to understand: what is the largest common divisor of two integers? Take the numbers 100 and 80, for example: what's the largest number that divides into both? You can make some assumptions about what that number might look like – it's going to be even and less that 40, obviously, and maybe more than 20 – but to get any closer is going to require a brute-force approach. Does 25 work? No. 30? Nope. Looks like it might be 20 then, as this divides into both and it doesn't look like there can be a higher number.

How about if the two numbers were 50 and 60? It's not obvious what the common divisor might be for these two, which introduces more guesswork. Or what if the numbers were 123456 and 654321?

Adding and subtracting

For all the non-Euclids, the most basic algorithm may simply halve the smallest number and then start counting down, checking whether the new number divides into both. It will work OK for small values, but it's obviously a computationally expensive approach that will become unrealistic very quickly. There has to be a better way, and that's where Euclid comes in. Euclid discovered that if you compare the smaller number with the difference between the smaller and the larger number, 50 compared to 10 in our second example, and then carried on doing the same comparisons, smaller compared against the remainder of the previous subtraction, until you could continue no further, the previous remainder is the largest common divisor. For the numbers 50 and 60, here's what happens:

> "We're going to use Python, for this, as it's installed on virtually everything"

60 - 50 = 10
50 - 10 = 40
40 - 10 = 30
30 - 10 = 20
20 - 10 = 10
10 - 10 = 0

So the largest common divisor between the numbers 50 and 60 is 10. Try it for yourself. It may get a little longwinded – you could easily see that the solution was going to be 10 in the previous example, for instance, but it always works regardless of how big the numbers are you choose. The next question we should be asking is, why? The solution is to do with common divisors, the group of numbers that can be equally divided into both of our values. The common divisor of **a** (assuming **a** is largest number in the pair), is also a common divisor of **a - b** (assuming **b** is the second number). In the first line of our previous calculation, that's the number 10 (60-50). 10 has its own set of devisors – 1,2, 5 and 10, and this process of subtraction doesn't change the set of common divisors. This makes sense because when you subtract the difference you are subtracting a number that shares the common divisors of both numbers.

It might help if you think about this in terms of reversing the calculations with addition:

10 + 10 = 20

20 shares the common divisors of 10, because we've just doubled it.

20 + 10 = 30

Each addition sharing the same common factor that we started with, until...

50 + 10 = 60

We now have our original two values, and you can see where the common divisors came into the equation and how the reversal of this reveals them.

The next job is to put this idea into code, and you should be able to see that we're on the verge of replacing our numbers with variables anyway, so we just need to add some logic. We're going to use Python for this example, as it's installed on virtually everything – from the Raspberry Pi to Apple's OS X and your Linux distribution. If you've not used the Python interpreter before, just type **python** on the command line and make sure you follow our syntax and indentation exactly. Here's the Python code:

```
def euclid(a, b):
        return b and euclid(b, a%b) or a
```

Woah! Those two lines of code do what we've just spent 700 words trying to explain!

If this is your first foray into Python, we'll try to take it as slowly as we can, starting off with what we've just created. **def euclid(a, b):** defines a function called **euclid** that takes two arguments: **a** and **b**. These values are the same two values we were using before in our explanation. If you've just typed this into Python, you can type **euclid(100,140)** to execute the function.

```
euclid(100,140)
```

The interpreter will spit out the answer, which in this case is 20. Now let's look at what the function is doing, one word or character at a time. **return** is how functions are halted when retuning results from an evaluation. If this line were **return 1234**, the output from the function would always be 1234. But that doesn't include any evaluating, which in our example, is done with the remainder of the line. The next character is **b**, our second number, followed by the word **and**.

Boolean operators

In programming terms, **and** is a Boolean operator. With most other programming languages, for an evaluation to be true both sides of a Boolean **and** need to be non-zero. (1 and 1) is true, for example, whereas (0 and 1) is false, and those languages would typically return a 1 for true and a 0 for false. Python is slightly different in the way it handles return values because it packs more features into a single operation. If the first value is non-zero, it will return the second value from the evaluation. If it's false, it will return the first. Here's a simple function definition and the output from the interpreter to show you what we mean:

```
>>> def andtest(a,b):
...  return a and b
...
>>> andtest(1,2)
2
>>> andltest(0,2)
0
```

This facility gives you the same output you get from other languages – if both values are non-zero, you'll get a non-zero value returned, which is effectively the same as **(1 and 1) = true**. If either the first or the second values are zero, these will be returned, effectively making **(3 and 0) = false**.

But you get more because you get the value of the second number for free, and this is how our code is working. But there's another trick immediately afterwards – recursion:

```
euclid(b, a%b) or a
```

The second argument to the first **and** evaluation calls the function again from within itself. That's the recursion part. The arguments for this second call of the function are the second value itself and the remainder of a division between the first and second number. This remainder of a division, otherwise known as a modulo operation, is a different method to the one we outlined earlier. It's the same theory, only made more efficient. This is because equal divisions of the lower number into the higher number – such as 5 into 28 – help us to fast forward a few steps without losing the common divisor. 28%5=3, which is because 28 divided by 5 = 5, with a remainder of 3. You get the same result as the remainder from the subtractions we were doing earlier, only without all the effort:

28 - 5 = 23
23 - 5 = 18
18 - 5 = 13
13 - 5 = 8
8 - 5 = 3

But when will this recursion stop? When will the function stop calling itself and start returning values back up the chain? That's where the final **or a** comes into play, and it's an evaluation connected to the earlier **and** statement. In most programming languages, an **or** evaluation will only return true if one or the other of the arguments is true – so **(1 or 0)** would equal true, but **(0 or 0)** would be false. In Python, you get better value from the same statement because it returns the first value if it's false and the second value if its not. Here's another quick example from the interpreter:

```
>>> def ortest(a,b):
...  return a or b
...
>>> ortest(1,2)
1
>>> ortest(0,2)
2
```

If the evaluation of the recursively embedded function returns zero, the **and** evaluates the value of **a** against the value of **b**, effectively returning the next to last value for **b** before the final evaluation returned **0**. That's exactly the same result we got when we first worked out Euclid's algorithm manually, but it's quite difficult to imagine. To make things clearer, here's some pseudo code for what happens when we call the function with the values of 60 and 50, showing each recursive step on a line with a number and the values Python is calculating. When a value is finally returned, we change the line number with the returned value inserted into the evaluation so you can see what's happening and how we step back through recursion to the final number:

```
a = 60 b = 50
1: 60 and euclid(50, 10) or 60
2: 50 and euclid(10, 40) or 50
3: 40 and euclid(10, 30) or 40
4: 30 and euclid(10, 20) or 30
5: 20 and euclid(10, 10) or 20
6: 10 and euclid(10, 0) or 10 (RETURNS 10 )
5: 20 and 10 or 20 (RETURNS 10)
4: 30 and 10 or 30 (RETURNS 10)
3: 40 and 10 or 40 (RETURNS 10)
2: 50 and 10 or 50 (RETURNS 10)
1: 60 and 10 or 60 (RETURNS 10)
```

You can test the logic of that comparison yourself without the recursive element:

```
>>> def eval(a,b,c):
...  return a and b or c
...
>>> eval(20,10,20)
10
```

The end result is the product of thousands of years of thought – a concise algorithm that performs a useful operation, all on a single line, while at the same time teaching a little about how Python maximises functionality with its Boolean operations (and also makes itself quite difficult to read in the process).

LINUX VOICE TUTORIAL

SOLVE WORD PUZZLES WITH BASH

BEN EVERARD

The humble command line interface is amazingly powerful, for both real work and playing games.

WHY DO THIS?

- Get to grips with egrep and extended regular expressions
- Never get stuck on word puzzles again
- Search through all the text files on your system with ease

It's no secret that Bash, the shell on most Linux systems, is an incredibly powerful tool, however it's one that many Linux users don't take the time to fully learn. A lot of tutorials focus on boring but practical uses like managing log files, but it doesn't have to be this way. Bash can be fun.

Here at Linux Voice, we want to give this tool some love, so we're inaugurating the Grep Games. This is an event where you use Bash together with grep to solve the sort of word puzzles you find in glossy magazines.

Here's an example: what is aedh an anagram of? To solve this, you're going to need a list of English words. This comes as standard on most Linuxes, and can usually be found at **/usr/share/dict/words** or **/usr/dict/words**. If it's not there, check for a **words** or **wordlist** package in your package manager. Failing that, you can grab it from the DVD or **linuxvoice.com**. In this article, we'll use **/usr/share/dict/words**, but you should change this if your **words** file is elsewhere.

We'll use egrep (like grep but uses extended regular expressions, which have a cleaner syntax than plain regular explessions) to find the right words. If you haven't come across this tool before, take a look at the boxout on grep and regular expressions, right.

You can find any word that contains just the letters aedh with this line:

```
egrep "^[aedh]*$" /usr/share/dict/words
```

The **^** matches the start of the line, **$** the end of the line and **[aedh]*** matches any string of the letters aedh. However, these aren't all anagrams. Any anagram must be exactly four letters long, so let's only match words of exactly four characters:

```
egrep "^[aedh]{4}$" /usr/share/dict/words
```

This is a bit better, but there are still some with repeated characters. To solve this we're going to pipe the output into a second instance of egrep, like this:

```
egrep "^[aedh]{4}$" /usr/share/dict/words |  egrep -v "(.).*\1"
```

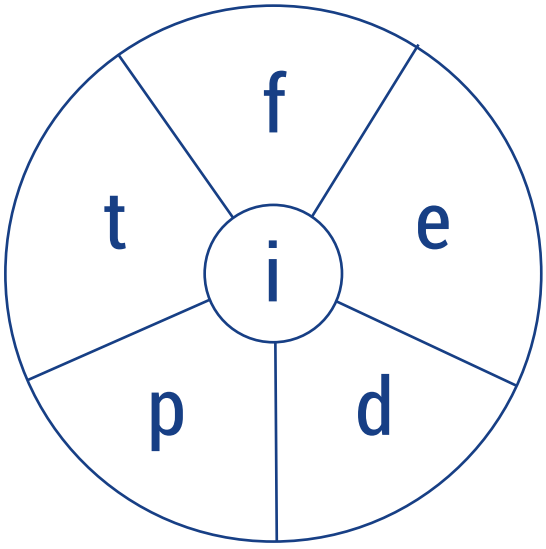
If you run this, you'll find that it only returns one line, the anagram of aedh. The second **egrep** has the **-v** flag, which means that it works in reverse; that is, it only outputs lines that don't match the pattern. The pattern **(.).*\1** matches any line with a repeated character in it because **(.)** matches any character, **.*** matches any string of any length (including nothing) and **\1** is a back reference to the first character. For more details on this, see backreferences in the boxout on Grep and regular expressions.

Sometimes an anagram will contain a repeated letter, and that would be missed by the above. Take, for example, **eeeddh**. The previous method won't work, so instead we need to match different letters different numbers of times. The code for this is:

```
egrep "^[edh]{6}$" /usr/share/dict/words | egrep  "*^[^e]*(e[^e]*){3}[^e]*$" | egrep "^[^d]*(d[^d]*){2}[^d]*$" | egrep -v "([^ed]).*\1]*"
```

Here the second and third egreps both work in the same way. They make sure that a particular letter is repeated exactly a certain number of times. **[^e]** matches any character except **e**, so the second egrep matches any string that starts at a new line, has any character other than a letter 'e' zero or more times followed by three occurrences of the bracketed expression (which contains e once and any string of other characters), then anything that isn't an e zero or more times followed by an end of line.

The final egrep makes sure that nothing other than e and d are repeated.



Word wheels: a challenging mental puzzle or a simple command?

I'll have a vowel please Carol

This solves complete anagrams, but that's not always what you want to do. In the UK there's a quiz show called Countdown, in which the contestants have to make the longest word they can out of a given sequence of nine letters.

You can solve this in a similar manner to the above problem, but by using ranges for the number of characters rather than an absolute number. Take a look at this example for the letters a,e,e,f,d,m,t,t,i

```
egrep "^[aefdmti]{1,9}$" /usr/share/dict/words | egrep "*^[^e]*(e[^e]*){0,2}[^e]*$" | egrep "^[^t]*(t[^t]*){0,2}[^t]*$" | egrep -v "([^et]).*\1]*"
```

However, this doesn't quite solve our problem. We don't want all the words that match, just the longest one. To get this, we need to go beyond a single line and create a script.

```
#!/bin/bash
longestLength=0
longestWord=""
while read word
do
 if (( ${#word} > longestLength ))
 then
   longestLength=${#word}
   longestWord=$word
 fi
done
echo $longestWord
```

This code reads each line from standard in (while read line) and checks its length against the previous longest word. At the end, it echos (prints) the longest word its found. To include this with the previous egrep commands, just use:

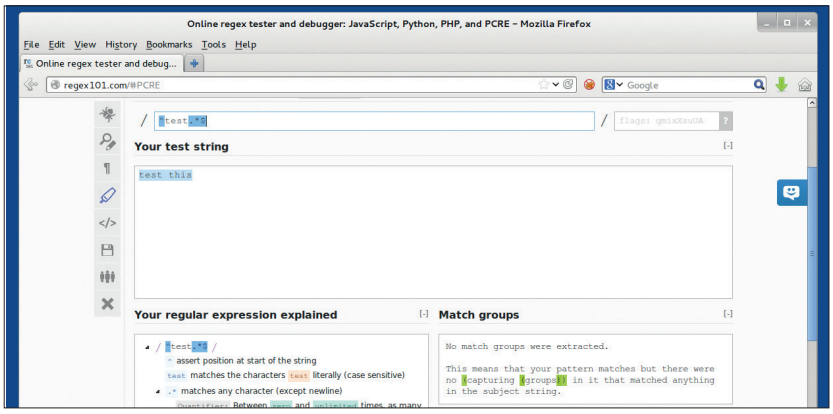
```
egrep "^[aefdmnti]{1,9}$" /usr/share/dict/words | egrep "*^[^e]*(e[^e]*){0,2}[^e]*$" | egrep "^[^t]*(t[^t]*){0,2}[^t]*$" | egrep -v "([^et]).*\1]*" | bash longest.sh
```

Where **longest.sh** is the filename of the above script (it's on the website and DVD).

Another puzzle similar to Countdown is the word wheel. This is where there's a series of letters on the outside of a circle and one in the middle. You then have to find as many words as possible that contain the letter in the middle and two or more of the letters on the outside. The example puzzle on the facing page can be solved with:

```
egrep "^[fedpt]*i[fedpt]*$"  re/dict/words | egrep -v "(.).*\1" | egrep ".{3,}"
```

Word ladders are a bit different to the puzzles we've looked at so far. Instead of arranging various letters into words, you start with a word, then each rung of the ladder you change a single letter from the word above until you end up with a final word.



www.regex101.com is an online tool to help you understand regular expressions. Unfortunately it uses regular expressions from PHP, Python and JavaScript, which are slightly different from egrep.

There are two separate parts to look at. The first part is finding all the words that can follow a particular word. The second part is finding out if a particular word can precede the final word.

Let's try the ladder:

```
live
----
----
----
raft
```

To solve this you have to come up with three words.

```
#!/bin/bash
for x in $(egrep "^liv.$|^li.e$|^l.ve$|^.ive$" /usr/share/dict/words)
do
    query='^.'${x:1:3}'$|^'${x:0:1}'.'${x:2:2}'$|^'${x:0:2}'.'${x:3:1}'$|^'${x:0:3}'.$'
    for y in $(egrep $query /usr/share/dict/words)
    do
        query2='^.'${y:1:3}'$|^'${y:0:1}'.'${y:2:2}'$|^'${y:0:2}'.'${y:3:1}'$|^'${y:0:3}'.$'
```

Grep and regular expressions

Grep is a popular tool for finding particular pieces of text. As well as solving word games, it's also useful in finding particular messages in log files and other 'real' work. egrep is like grep, but it uses extended regular expressions rather than ordinary regular expressions. These have a cleaner syntax, so it's these that we'll use here.

The basic usage is:

```
egrep <pattern> <file>
```

This will output every line in the file that matches **<pattern>**. It can also be used in a pipe like this:

```
cat <file> | egrep <pattern>
```

This just prints every line that **cat** outputs that matches **<pattern>**.

The trick with egrep is in mastering extended regular expressions.

A letter just matches itself, so for example, **abc** will match any line that contains the string abc anywhere in it. **^** matches the start of the line and **$** matches the end of the line, so **^abc** matches any line that starts with **abc**, **abc$** matches any line that ends with **abc** and **^abc$** matches any line that contains just **abc**. The "**.**" character matches any character, so **^a.c$** will match **abc**, **adc**, **aac**, but not **ac**. This is known as backreferencing.

You can also match groups of characters, eg **^[ab]** will match any line that starts with **a** or **b**, while **^[^ab]** will match any line that starts with any character other than **a** or **b**. **^[a-z]** will match any line starting with a lower-case letter. There are also a few special options here such as **[[:space:]]**, which matches any whitespace (space, tab, etc) and **[[:lower:]]** which matches any lower-case letter.

You can match characters more than once. ***** matches zero or more times, **+** one or more time, and **?** zero or one time. So, **^a*$** matches a line that contains a number of **a**'s but no other characters. **^a.*a$** matches a line that starts and finishes with a letter **a**. **^a.+a$** matches any line that starts and ends with an **a** and has at least one character in between. You can also specify a range of the number of matches you want by using **{}**. For example, **^a{2,3}$** will match the lines **aa** and **aaa**, but nothing else. You can bracket parts of regular expressions as well. This is useful because it allows you to refer to particular matches. **\1** matches whatever the first bracketed expression matched, **\2** matches what the second matched and so on. For example, **(.)\1** will match any two characters that are the same separated by a character, such as **bob**, **did**, **aaa**, but not **abc**.

The final part of extended regular expressions that we'll look at is **|**. This allows you to match against more than one pattern. For example, **^ab|^bc** will match anything that starts with either **ab** or **bc**, but not **ac** or anything else. **^(ab|bc)** does the same thing.

```
        for z in $(egrep $query2 /usr/share/dict/words | egrep "^raf.$|^ra.t$|^r.ft$|^.aft$")
        do
                if [ $x != $y ] && [ $x != $z ] && [ $x != "live" ] && [ $x != "raft" ] && [ $y != $z ] && [ $y != "live" ] && [ $y != "raft" ] && [ $z != "live" ] && [ $z != "raft" ]; then
                echo "live"
                echo $x
                echo $y
                echo $z
                echo "raft"
                echo "---"
                fi
        done
    done
done
```

This code performs three for loops, one for each of the missing words. The first for loop runs on every word that matches the regular expression **"^liv.$|^li.e$|^l.ve$|^.ive$"** this is effectively four different regular expressions separated by **|**. Together, it will return any word that matches any one of these sub-expressions.

Inside this for loop it runs the line

```
query='^.'${x:1:3}'$|^'${x:0:1}'.'${x:2:2}'$|^'${x:0:2}'.'${x:3:1}'$|^'${x:0:3}'.$'
```

This just builds up a regular expression equivalent to the first one but for every word returned. **x** is the variable holding the word, and **${x:1:3}** (for example) returns characters 1 through 3 of the word held in variable x (the first character is 0). The second for loop works in exactly the same way as the first.

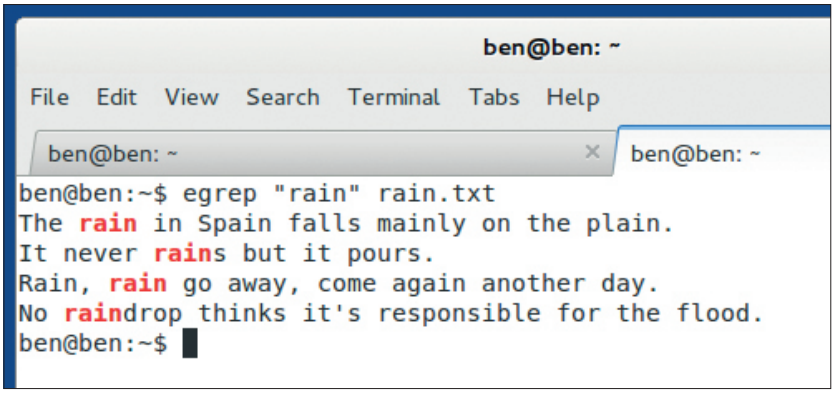
The final for loop is a bit different because it not only has to match the word above it, but the word below it as well. For this reason it runs two egreps on the words: one to match the words above, and the second to match the words below. The if statement simply removes any solutions that repeat words.

Playing GCHQ

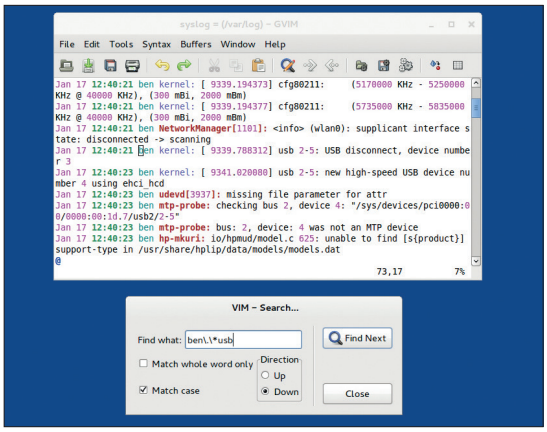
Substitution ciphers are easy-to-break encryption systems where you take each letter of the alphabet and represent it with a different symbol. The point of the puzzle is to work out what letters the symbols represent. As an example, the cipher:

```
12334, 56 7852 90 a27
```

could correspond to:



egrep will highlight the particular part of each line that matches the regular expression.



Many programs have some form of regexes built in. Here, gvim is finding all USB messages for user ben in the syslog.

```
hello, my name is ben
```

because h=1, e=2, l=3, o=4, m=5, y=6, n=7, a=8, i=9, b=a.

Now take a look at the following:

```
123452 672 8298a2 bc 9889dbeb9c
```

The main clue here are repeated letters which you can match using back references. You could try to build a script to match the whole lot in one go, but it's far easier and quicker to pick on part with quite a few repeated characters and just match that. Once you've got that, it should be quite trivial to finish it off. We decided to work with the final two words. A script to solve them is:

```
#!/bin/bash
list2=$(egrep "(.)(.)\2\1.(.).\3\1." /usr/share/dict/words)
for word1 in $(egrep "^.{2}$" /usr/share/dict/words)
do
        for word2 in $list2
        do
                echo $word1" "$word2 | egrep "^(.)(.)[[:space:]](.)(.)\4\3.\1.\1\3\2$"
        done
done
```

The first loop goes through every two letter word while the second one loops through every word that matches the particular pattern of backreferences.

The guts of the code is the line:

```
echo $word1" "$word2 | egrep "^(.)(.)[[:space:]](.)(.)\4\3.\1.\1\3\2$"
```

It checks every pair of words generated by the two wloops for a particular pattern of back references which correspond to repeated characters in the ciphertext.

This method could be expanded to match three or more words, though it will slow down significantly with each new word.

Once you've got some of the letters, you should be able to come up with patterns based on the letters you know to find the other words.

Ben Everard is the co-author of *Learning Python with Raspberry Pi*, soon to be published by Wiley. He's also pretty good at turning foraged fruit into alcohol.

Challenges

Test your skills by writing scripts to solve the following word puzzles

Anagrams

ainpprss
abeprrrsy
bbceirssu

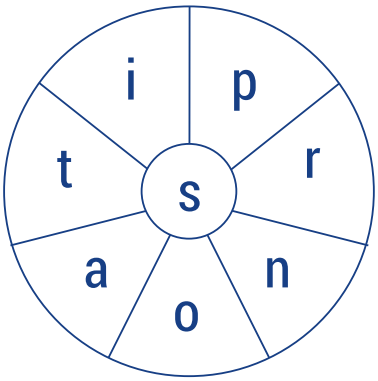
Countdown

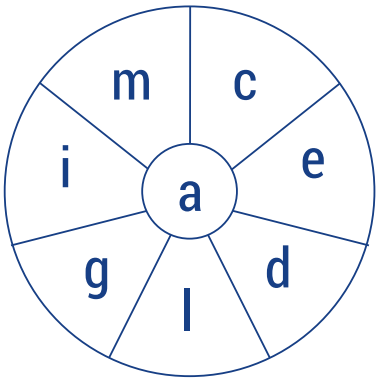
tnxpamies
dimtescat
hofanescp

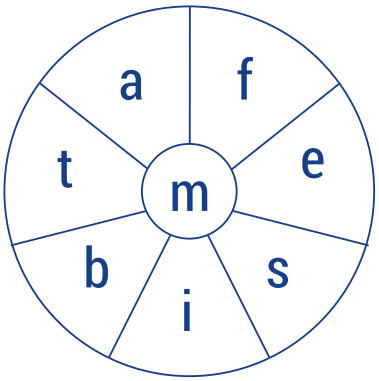
Encryption

1 2134 567894550 518824 1a4 a546b4
1234 34 5641 127 879300309
123 456 4 378936 8708a8034b

Word wheel







Word ladder

band	**brag**	**wire**
----	----	----
----	----	----
----	----	----
meat	**plan**	**pant**

Find words, win clothing!

The final challenge is something different, one we haven't covered so far: a word search. To make matters a little easier, there are only horizontal words, and none of them are backwards. The challenge is to write a Bash script that can go in the following pipe:

```
cat wordsearch.txt | bash yourscript.sh
```

and output all of the three- or more letter words from the words list (**/usr/share/dict/words** or **usr/dict/words**) that are in the file.

The word search is (on the DVD and website as **wordsearch.txt**):

```
zfghellohb
binarytwno
thisenthat
dfjunglwmr
scoffeeqwj
lhzniphoto
rlightovqx
yelsocketn
fbicycleow
ykerolbuha
```

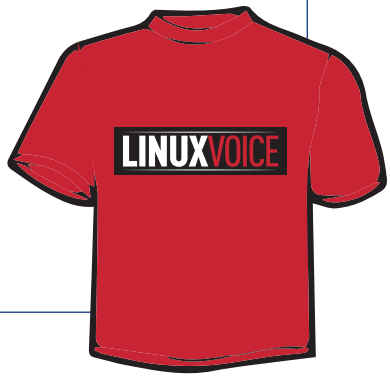
To make things interesting, there will be two prizes, one for the smallest (in terms of characters in the Bash script), and one for the one that runs fastest (completes execution with all the words found in the shortest time). The words must be found with a form of GNU **grep** (**egrep**, **pgrep**, etc.) matching a regular expression.

There are few differences between versions of Bash on different Linuxes, so we'll be testing on a fresh install of Debian Unstable. This is only likely to matter if you're relying on particularly new or exotic features.

To be eligible, your program must be licensed under an OSI-approved licence compatible with the GPL v2 or v3. We recommend using GPLv3.

All entries must be sent to **ben@linuxvoice.com** by 31 March, and the winner will be announced in issue 3 (and on linuxvoice.com). You don't have to have bought a magazine to enter (details will be posted to the website) so feel free to pass details of the competition on others.

In the event of a tie, the solution that was sent in first will win. In all matters, the editor's decision will be final.



LINUX VOICE

GET TO KNOW VIM

MASTERCLASS

Essential Linux tools explained – this month, say hello to the Vim text editor and some advanced features in Firefox

AN INTRODUCTION TO VIM

Whatever Unix system you're using, it will probably have Vim – let's learn this essential editor.

JOHN LANE

Vim, which is a hugely flexible text editor for Linux, is "Vi Improved". It carries a legacy that can be traced back to the 1970s, when Vi was born as the 'visual mode' for the Unix line editor, Ex (Ex itself was the eXtended version of the editor, Ed). The most jarring part of that legacy for new users is that Vim has two main operating modes.

LV PRO TIP

":help vim-modes" helps to explain Vim's operating modes

Command Mode is where you interact with Vim to move around, alter, save and exit; pretty much everything you can think of except the one thing you might want to do: type text into it. That's what the other mode (Insert Mode) is for. Sadly, many people take the time to learn one command at this point: **:q**, the quit command, so they can go off and find another editor. But *Linux Voice* readers are a hardy bunch, so we're staying with it...

Getting around

LV PRO TIP

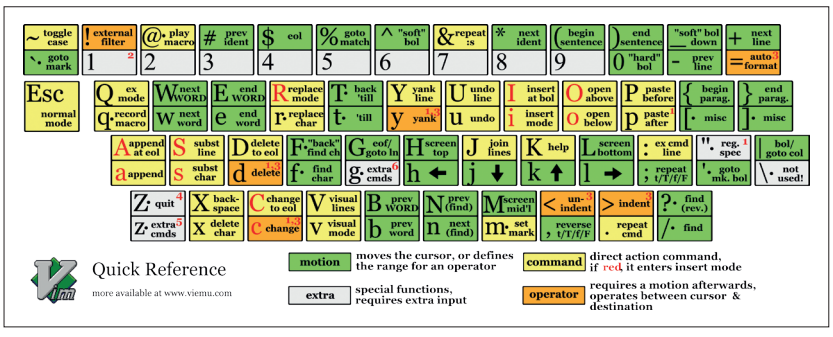
Most linux distros alias "vi" to "vim"

Another surviving part of its legacy is its quirky use of the **h**, **j**, **k** and **l** keys to move around the file. This can be traced back to Vi's creator using a terminal that didn't have separate cursor arrow keys but used these keys instead. Vim supports the normal arrow keys as well, but it's worth learning to use those letter keys because they're there directly underneath your fingertips and it'll make you much quicker once you've trained your muscle memory. For die-hards, the arrow keys can be turned off.

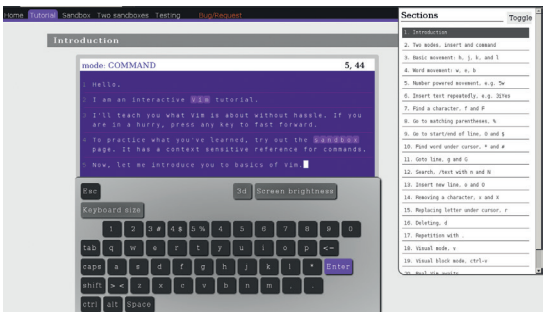
Quirks aside, let's get into Vim:

vim myfile

Like we mentioned, you'll find yourself in Command Mode. For basic navigation, you use the **hjkl** keys to move left, down, up and right within the file or there are commands that navigate words, sentences, paragraphs and pages.



Vim's Command Mode offers heaps of powerful commands to manipulate files.



Vim benefits from many online learning resources, like this interactive tutorial (www.openvim.com).

Commands are case-sensitive so **j** and **J** do different things. Most commands are one or two keystrokes and the first you'll want to learn is **i**, which enters Insert Mode. Once in Insert Mode, anything typed will be entered into the file until you press the Escape key, which returns you to command mode. Insert Mode is meant for short bursts of text entry – you're meant to place the cursor, enter Insert Mode, type some text and Escape back to Command Mode. New users might try to enter Insert Mode and then work as if they are using a simpler text editor such as Notepad, but to do so misses the point and power of Command Mode.

The next characteristic of Vim to understand is its ability to repeat commands. Any keystroke command given may be preceded with a number and that command will be repeated that many times. For example. enter **5h** to move left five characters. Using repeats like this with the navigation commands can help you move around your file very quickly.

Moving around is fine, but command mode also offers many ways to make modifications to your file. Begin with the basic (**i**)nsert, (**r**)eplace and e(**x**)tract. Some commands, like (**c**)hange and (**d**)elete, are

followed by a mandatory operator specifying what to operate on – these operators are the same as the navigation commands, so **3dw** deletes three words, **5x** deletes five characters.

A more sophisticated way to modify files is to use Ex commands. These hark back to the days before Vi when people used teletypes and didn't even have a monitor in front of them. Ex commands are typed at a prompt and perform modifications to the file being edited. You enter an Ex command from Vim's command mode by first entering a colon (**:**) and then the command. Ex commands take the form

```
:[range] command [args]
```

where the range and arguments are optional. The range defines the line, or lines, to perform the command on, and can be specified using line numbers, marks or special symbols: a full stop character represents the current line and **$** represents the last line. **%** is equivalent to **1,$**. You can create marks to represent lines of your choice: to define a mark, **a**, at your current position in a file, enter **ma** and you can then refer to that line as **a**. You can also use simple expressions like **+3** to refer to the line three ahead of the current line, or **$-10** to refer to the tenth-before-last line. Probably the most commonly used ex-command is to perform a global search and replace: **:s/%/foo/bar/g** replaces all occurrences of **foo** with **bar**. **foo** is a regular expression.

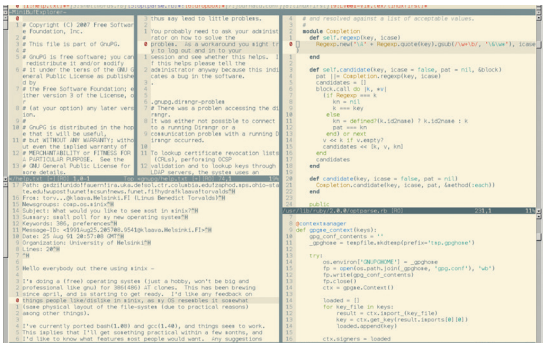
Vi has commands to cut or copy and paste text but it calls the copy action Yank instead. The commands are (**d**)elete, (**y**)ank and (**p**)aste. Deleting and yanking place the affected text into a general purpose buffer (which is not the desktop's clipboard). You can also use a named buffer by prefixing a command with **"** and a character (which names the buffer).

```
"a3dw
"byy
```

will delete three words into buffer **a** and yank the whole line into buffer **b**.

Buffers, windows and tabs

When Vim loads a file, it reads it into a buffer, which is the in-memory copy of the file that is altered as you edit it. The original file is only modified when you write the buffer back to disk. You use the **:w** command to do this. Vim can have many files open at the same time, each being held in its own buffer. You can list all buffers using **:ls** and navigate between them with **:b**. Split windows is a feature of Vim that enables you to view the file being edited in multiple places, or even view multiple different files. To split the window, enter **:sp** for a horizontal split or **:vs** for a vertical one. This will display the current buffer twice. Append a filename to instead load another file into the split window. If two windows display the same buffer, changes made in one will be reflected in the other. Window control commands are initiated with **Ctrl+W**. Moving between windows involves using **Ctrl+W** followed by a movement key, for example **Ctrl+W k** moves to the window above.



There are lots of ways to organise multiple files in Vim.

The other way to organise multiple files is to use tabs. You can have multiple tabs, each displaying multiple windows and, remember, each of those windows can display content from any buffer. Tab commands begin with **:tab**.

Pimp your Vim

One of the big features of Vim is that it is heavily customisable. You can view and change settings with the **:set** command. To see all settings:

```
:set all
```

And to enable a setting, for example:

```
:set number
```

will display line numbers. Prefix the setting name with **no** to disable it:

```
:set nonumber
```

To apply your setting preferences when starting Vim, you can store them in a file called **~/.vimrc**. This file can do more than just settings, though: by writing Vimscripts, you can create new functionality or modify existing features.

A good place to start customising is by key mapping. Any Command-Mode key sequence can be mapped to a key:

```
:map - dd
```

will cause Vim to delete the current line (that's the **dd** command) whenever you press **-**. Beware, however, mapping single keys will remap the main commands (there are no free single characters). To help avoid this you can use modifiers:

```
:map <c-d> dd
```

will map **Ctrl+d** to perform **dd**, which deletes the current line. You can also map multiple characters but you'll need to type them quickly to get their effect:

```
:map ,d dd
```

A coder's paradise

You can control whether tabs are hard or soft (implemented with spaces, often preferable these days). You can set up automatic indentation and code folding (hiding function bodies). The **=** command will indent the current selection. **:fold** will fold any selected block; **zo** opens a fold and **zc** closes it.

Finally, this tour would be incomplete without mentioning Gvim, which wraps Vim in a GUI window and exposes some of its commands through a traditional menu interface.

Do you want to know more?

- The Vim Tips Wiki **http://vim.wikia.com/wiki/Vim_Tips_Wiki**
- Open Vim Tutorial **http://openvim.com**
- ViEmu cheat sheet and tutorial **www.viemu.com**
- Vim website at **www.vim.org**

AN INTRODUCTION TO FIREFOX

Get more out of the flagship free software project's hidden features.

If statistics are to be believed, the Mozilla Firefox web browser enjoys around a 20% share of the browser market, lagging behind Google's Chrome browser, with its more polished look and feel. But Firefox offers more power-user features through its ability to be tweaked and customised.

The name Firefox (which is another name for the red panda) was chosen after earlier names, including Phoenix and Firebird were dropped due to trademark disputes. More on that later...

Mozilla releases Firefox versions at four stages in its lifecycle. Known as channels, they are:

- **Firefox Nightly**, a developer-focussed release aimed at "extremely technical" early adopters. It's updated every night and contains the latest changes.
- **Firefox Aurora** is a pre-beta release every morning for technical early adopters.
- **Firefox Beta** is a weekly beta release aimed at non-technical early adopters.
- **Firefox Final** is the official public release version intended for everyday, mainstream users. It is released on a six-week schedule.

Functional changes and enhancements move through the four channels, being initially available in the nightly build and moving through Aurora and Beta before eventually making it into a new public release. The release schedule is maintained on the Mozilla Wiki (**http://bit.ly/ffrelcal**).

In addition to the four main channels there's another version, called Firefox Extended Support Release, which is intended for large organisations such as universities, governments and businesses. Each ESR release is supported for one year, however it only receives high-risk/impact security fixes or major stability fixes.

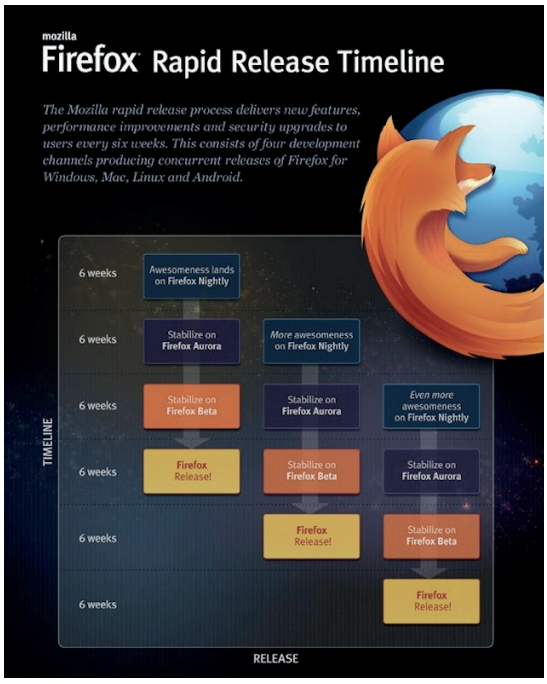
A quick tour

Central to the Firefox experience is what Mozilla calls the Awesome Bar, a combined URL and search box. You start typing what you want and the Awesome Bar displays possible matches from your browsing history, bookmarked sites and open tabs. When a page is displayed, its URL is prefixed with its 'favicon', the small icon used by some websites to give a brand identity to the URL. If you connect to a secure site, the identity information can be revealed by clicking on the favicon.

Bookmarking is quick and easy, thanks to the one-click bookmark feature accessed via the star icon at the end of the Awesome Bar. Click once to create a bookmark, click again to edit or remove one.

An extension to bookmarking is Quick Search. Say you frequently look up content on Wikipedia; if you browse to that site, right-click in its search bar and select Add A Keyword For This Search, you'll then be able to search it directly by entering **wikipedia my-search-term** in the Awesome Bar.

Firefox was one of the first browsers to offer the now-common tabbed browsing model. It extends the concept by enabling the user to pin certain tabs, meaning that they become permanent fixtures on the user interface where they space-efficiently display only the site's favicon on the left-hand end of the tab



Firefox's releases are fed through four release channels

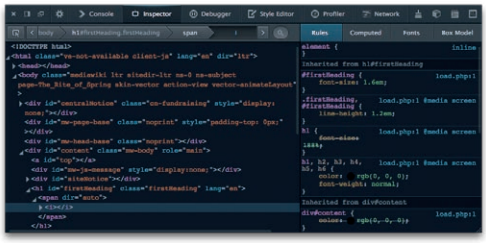
Our favourite add-ons

There are thousands of add-ons available through **http://addons.mozilla.org** and everyone will have their own favourites. Here are some of ours.

- **Pentadactyl** aims to make more efficient and keyboard accessible by bringing Vim-like keybindings and Ex-like commands to Firefox with a neat, minimalist GUI. This is a must-have add-on for Vim lovers **http://bit.ly/ffpentadactyl**.
- **Lightbeam** uses interactive visualisations to show you the first and third-party sites you interact with on the web.
- **AdBlock Plus** enables you to filter out unwanted parts of websites that you visit, such as noisy ad.
- **Abduction** lets you easily select an area of a web page with your mouse and save it as an image.
- **Firebug** is a web development tool that you can use in addition to, or instead of, the integrated developer toolbox.

Web developer tools

Web developers have another good reason to use Firefox. Its integrated Toolbox for web developers, accessible via the Tools > Web Developer menu, has improved over time, obviating the need for extensions such as Firebug. You can inspect elements and even make changes to HTML and CSS on the page that you are viewing – full details at http://bit.ly/ffwebtoolbox.



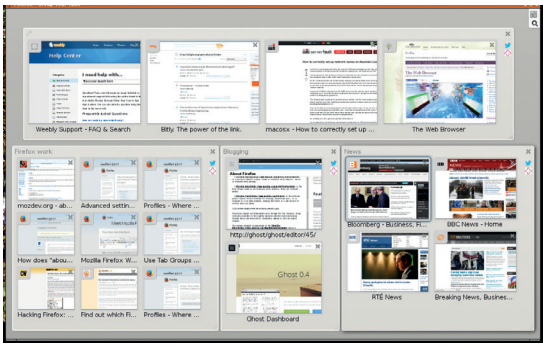
Firefox's Toolbox is great for web developers.

bar. They don't have a close button, thus avoiding accidental closure, and they glow blue when the page updates. Links on pinned tabs are opened in a new tab so that the pinned tab doesn't change (although beware, if you browse elsewhere while viewing a pinned tab, it changes the pinned tab!).

Manage your tabs

Tab addicts will like tab groups, (also known as a panorama). This is a great feature that you'll want to use if you often end up with more tabs than will fit on the tab bar, or if you want to organise tabs by task. To create tab groups, enter Ctrl+Shift+E or click the small icon at the right-hand end of the tab bar. This will display your tabs as thumbnails that you can drag around and collect into groups. You select a tab to display by clicking its thumbnail, which returns to the main display with that tab open and the tab bar only showing tabs from the same group.

Firefox is a feature-rich application but we can extend, bend and manipulate it to customise the browsing experience. This is made possible by its extendable architecture and by tweaking its internal settings. Everything you do to customise Firefox is



The panorama: ideal for users who like lots of tabs.

stored within a profile that you'll find under your home directory in a path similar to

/home/your_user_name/.mozilla/firefox/k9uvtyeq.default/

The exact path contains a random string that serves two purposes: it makes the profile directory unique and obfuscates it to reduce of the risk of malware locating it on platforms where that kind of thing is an issue. Most users will have one profile but it is possible to have many and you may find this useful if you are experimenting with extensions or settings. To find the profile directory that Firefox is using, you can browse to **URL:about:support** and click the Open Directory button, which is next to the Profile Directory label.

Settings are stored in your profile in a plain-text file called **prefs.js**. You can look at this file in any text editor but changes are best made using the application. There are many more settings than you can access via the Preferences dialog – you instead enter **about:config** into the URL bar, whcih will bring up a huge range of preferences to tweak (if you accept the risk of potentially breaking things, of course!).

> **"Firefox was one of the first browsers to offer the tabbed browsing model."**

To set an option, search for it and click the value to change it. For an example, search for 'animatefade' and change its value from the default **false** to **true**, to get some fun eye candy.

Privacy and security

The awesomeness of the Awesome Bar, and other things, require Firefox to collect data about your browsing habits. But you have control over this. One way is to use Private Browsing, which enables you to browse the internet without saving any information about which sites and pages you visit. It doesn't make you anonymous on the internet, however. The keystroke Ctrl+Shift+P starts private browsing in a new window (a small mask icon to the left of the tab bar identifies it as being private). Firefox won't track browser, search, download and web-form histories, cookies, or temporary internet files (any files you specifically download or bookmarks you make will be kept). You close the window to stop Private Browsing.

Firefox can tell the sites that you visit that you do not want to be tracked, however it doesn't do this by default so you should go to **Preferences > Privacy** to select this option. There is, however, no requirement that web sites honour this setting. An extension called Lightbeam can reveal the sites that are tracking you online. Privacy is a balancing act: read the Mozilla page at **http://bit.ly/ffprivacy** to get the current Firefox privacy policy.

LV PRO TIP

Start with Firefox from the command line with **firefox -P** to manage profiles.

John Lane is a technology consultant with a penchant for Linux. His Firefox has a penguin theme.

THANK YOU!

Linux Voice would like to say a huge thanks to everyone who made this magazine possible. If you bought the Founder perk during our IndieGogo campaign, your name should be somewhere on these pages. You're awesome!

LINUX VOICE

The open source magazine that gives back to the Free Software community

Adam Hawton, Adrian Busolini, Ahmad A. J. Fallatah, Alan Shuttleworth, Alan Underwood, Alberto Goldoni, Alberto Mardegan, Aleksey Shumanev, Alessandro Franzì, Alex Bishop, Alex Thomas, Alexander Orr, Alf Orseni, Allen Antonio Dawson, Alphonsos Pangas, Andrea Gallo, Andreas Zweili, Andrei Sebastian Cîmpean, Andrew Allen, Andrew Bowden, Andrew Grant, Andrew Howe, Andrew Perkins, Andrew Walker, Andrew William Hearn, Andrew Wilson, Andrew Wozniak, Andrew Yeomans, Andy Baker, Andy Church, Andy Vickers, Anna Smith, Antti Aspinen, António M P Mendes, Areski Belaid, Arturo Fernandez Montoro, Arturo Hoffstadt Urrutia, Asavar Tzeth, Ashley Manly, Ashley Seabrook, Astiak Ahmed, Austin Prior, Axel Sommerfeldt, Barry Hughes, Barry Polley, Bart Van Melckebeke, Beardy Jesse, Becky Newborough, Ben Bryant, Ben Oliver, Bill B. [SuperEngineer], Bill Quinn, BitFolk, Bjarni Petersen, Bjorn_G, Björn Wijers, Blaise Alleyne, Blake Johnson, Bob Moss, Boele Ruurd Kuipers, Brian Klump, Brian McLaughlin, Bruce Beardall, Bruno Rodrigues, Bryan Andrew Butler, Bård Aase, C. Victor Pollard, Callum Scott, Campbell Barton, Carl Ryan Davies, Carla Sella, Catherine Morrison, Ceśar Fernández Rodríguez, Charles Forbes, Charles Kerr, Charles Lees, ChaseFox, Chris Bishop, Chris Bland-Ward, Chris Chapman, Chris Dell, Chris Ellis, Chris Lee, Chris Sharpe, Chris Tipney, Christian Christiansen, Christian Dannie Storgaard, Christian Lange, Christof Kluß, Christoph Stenglein, Christopher Brandon, Christopher Sandles, chris_debian, Claes Bergman, Clive Bull, Clive Rixson, Colin Haynes, Colin Marshall, Colin S. Hamilton, Colin White, Connie New, Craig Allen, Craig Ewers, Craig Waites, Cryptable (able to use crypto), Curtis Adkins, Cyryl Płotnicki-Chudyk, Dafydd Young, Dale Malony, damo, Daniel Alan Miller, Daniel Egger, Daniel Harlea, Daniel Llewellyn, Daniel Parraz, Daniel Venema, Darryl Weaver, Dave Cross, Dave Harris (G0CER), Dave Kelly, Dave Lewis, David 'Peanut' Peter, David Anderson, David Baxter, David Herel, David J. Meier, David Lewis, David Mugnai, David W. Lee, David Waters, Denis Moroz, Derek Smit, Dick Bain aka towy71, Don Brown, Don Newell, Drakonas, Dustin Sysko, Ed McDonagh, Ed Vipas, Einar Petersen, Eion MacDonald, Emanuele Rogledi, Emil Hemdal, Emmanuel Rialland, Eoghann Walker, Eric Crampton, Ernie Hutschemaekers, Espen Klein Nilsen, Fay Henderson, Francesco Cervellera (Ciscoski), Frank Bell, Frank Bonner, Frank Fegert, Frank Jung, Frank Thomas-Hockey , Franz-Robert van Vugt, Fredrik Fyksen Lund, Fusion Ticket Solutions, Félim Whiteley, Gareth Thomas, Gary Walker, Gary Wilson, Geoffrey Robertson, George Angelopoulos, George Tripp, Gio Ciampa, Giovanni Moretti, Giovanni Panozzo, Glen Coulthard, Glenn Holmer, GLITTAH, Gordon O'Brien (MSQ Partners Ltd), Graham Cobb, Greg Wellington, Gwilym Kuiper, Géza Piros, H. de Wind, Hans-Georg Eßer, Hoyt Duff, Hugo Gonçalves [vyanix], Iain Cuthbertson, Ian Fox, Ian Harcombe, Ian Hatton, Ian Lowry, Ian Patterson, Ian Price, Igal Alkon, Immo Zadezensky, Intech Fusion, Iván Tomé, Jack (jfk) Knight, Jacob Kiers, Jacopo M. Colucci, James

Beal, James Derrick, James Evans, James Seymour, Jan Henkins, Jan Kudrik, Yann Chx, Jason Daborn, Jason Duke, Jason King, Jason Pollitt, Javier A. Ochoa M, Jean Azzopardi, Jean Visconte, Jean-Michel Lavarenne, Jeffrey Honig, Jens Klün, Jens Skovgaard Jensen, Jens Weise, Jeppe Nielsen, Jeremie Gobeil, Jeremy DeBlois, Jeremy Roe, Jeremy Schrock, Jeroen Franssen, Jesús Alonso Fernández, Jiggy Kakkad, Joel Jorgensen, Johan Bergman, John, John and Anne Everard, John Craick, John de Mattos, John Dean, John Fraser Williams, John Gilbert, John Harrold, John Holbrook, John Hunter, John Mertz, John Neese Jr, John Uren, Johnbuoy, Jolyon T Hallel, Jon "The Nice Guy" Spriggs, Jon Gibbins, Jon Peder Horn Opsahl, Jon Ware, Jon Wilks, Jonatan Bijl, Jonathan Brown, Jonathan Monahan, Jonny McCullagh, Joona Lehtomäki, Jorge Fábregas, Jorma Turkenburg, Joseph VanPelt, Josh McKenzie, Joshua Mills, José Venceslau, Jouni Kvist, Juanjo Marin, Julian Beach, Jørn Ølmheim, Keith Pawson, Kenneth Geisshirt, Kev Quirk, Kevie (TuxJam), Kevin "Velkro" Watret, Kevin B. O'Brien, Kevin Colyer, Kevin R Soulsby, Kevin Ramsay, Kevin Safford, @kimondo, Kristian Tizzard, Kristof Van Eyken, Kyle A. Matheny, Lachlan Rhodes, Larry Bain, Lars Hagström, Laurence Cable, LDWatson, Lee Moffitt, Lee Stone, Lee Trager, Leo Leibovici, Louis Bohm, Luis Gonzalo Fierro, Luis Viant, Lukas 'Spydon' Klingsbo, Lukas Ahrenberg, Lukas Mutschmann-Sanchez, Lynn Griffin, Lynn Root, madwoollything, Magne Djupvik, Manpreet S. Shinhmar, Manuel A Ortega Jr, Marco Trevisan, Marek, Marius Pfeifer, Mark Baker, Mark Cohen, Mark Craig, Mark Diamond, Mark Holmes, Mark Niski, Mark Skinner aka CaptainMark, Mark Wolf, Martin Buhr, Martin Cooper, Martin D Turner, Martin Watts (The White), María García Miranda, Mass Dosage, Matt Bynum, Matt Morgan, Matt Tei, Matthew Cronen, Matthew Hudson, Matthew L. Weber, Matthew Mellor, Matthew Stevenson, Matthew Tunstall, Matti Lammi, Mattias Ohlsson, Maurice George, Max Dowkes, Max Eaves, Max L, Maxim Andersson, Michael Ashe, Michael Coyne, Michael Noyce, Michael O'Neill, Michael O'Sullivan, Michael P. Burton, Michael P. Fumicello, Michael Rimicans, Michael Rose, Michael V. Gioia, Michael Whitmore, Micke Eklund, Mika Impola, Mike Ball, Mike Bishop, Mike Bright, Mike Hingley, Mike O'Day, Mike Sander, Mike Wort, Mikkel Kristiansen, Milos Gajdos, MintyfreshPenguin, Mårten Behm, Nathan Barnes, Neil Grogan, Neil Mark Simmons, Nick Crawford, Nick Fox, Nick Heppleston, Nick Prescott, Nicola Mazbar, Nicolai Abruzzese, Nigel Green, Nigel Metheringham, Nigel Small, Nikos Roussos, Norman Lorrain, Ole-Morten Duesund, Ortega, Oskar Hannesson, Owain Clarke, Owen Savill, ozten, Pat McClung, Pat Riehecky, Patrick A. TenHoopen, Patrick Guido Arminio, Patrick Latour, Patrik Jeppsson, Paul Dann, Paul Holman, Paul Keeton, Paul Smedley, Paul Walker, Paul Weeden, Paul Whelan, Paul Whittaker, Paul Williams (heiowge), Paw Hermansen, Per-Erik Westerberg, Pete Crilly, Pete Martin, Peter Bradley, Peter Carl, Peter Hultqvist, Peter Jan Huls, Peter Knowles, Peter Malaise, Peter Vachuska, Phil Robinson, Phil The Sheep, Philip Davidson, Philip Whitaker, Philippe Le Toquin, Phillip Horger, Pi Supply UK, Przemysław Kamiński, Raf Van Opdorp, Ralph Corderoy, Randall Crook, Rasmus Wriedt Larsen, Ray Foulkes, Remy van Elst, Renzo Massari, Rich KD0RG,, Rich Wareham, Richard C. Dawson, Richard Delaney, Richard Lee Buff, Richard Waterman, Ricky Spanish, Rob Gilijamse, Rob Probin, Rob Shearer, Robert Cook, Robert Gravsjö, Robert J. Collins, Robert Longstaff, Robert M. Albrecht, Robert Meineke, Robert Mundal Axelsen, Roger J A Duthie, Roger McNally, Ross Brunson (LPI), Rouven Hernier, Ruben Manukyan, Rui Gouveia, Rusty Ramser, Ruud Schellekens, Ryan Hartlage, Sadie Simpson, Sam Kirby, Sami Flew, Saqib Razaq, Sarapis, Sava Slijepčević, Scott Buffington, Scott Fishman, Scott Meikle, Scott Newlon, Seumas Mackinnon, Shaun Walker, Shelby Cruver, Shiela Everard, Sigg3.net, Simon Bell, Simon Given, Simon Handy, Simon Quinton, Simon Spice, Spencer Hunley, Stacy D. Uden, Starfinger, Starfire Information Technologies, LLC, Stefan Beller, Steinar Wenaas, Stephen Kirkby, Stephen Lang, Stephen Piggott, Stephen Riddett, Steve Bickle, Steve Perry, Steve Taylor, Steve Tremayne, Steve van Rossum, Steven Bianchi, Steven Le Flohic, Stewart Robertson, Stuart Ward, Svend Gundestrup, Svenning Andersson, Svetlozar Argirov, T. Tonetti (licht_t), Tadeusz Cantwell, Tanel Mae, Teodor Gambal, Terence S. J. Sambo, Terry Dooher, The Sunlay Corporation, thelovebug, Thierry Nicola, Thomas Blood, Tim Abell, Tim Elliot, Tim Foxcroft, Tim Hamilton, Tim O'Connell, Tim Speetjens, Tod Davis, Tom da pud, Tom Lange, Tom Larard, Tom Parrott, Tomasz Muras, Tommi Laukkanen, tommybobbins, Ton Sonneveldt, Tony Hughes, Tony Wood, Tryggvi Björgvinsson, Tyson Moore, Venomenus, Wayne Rooney, yajupadu, Yeu Sian Lee, Yury Vidineev, Zoë Williams

LINUXVOICE DVD 001

Distros, videos, applications, games, podcasts and more...



A NUGGET OF PUREST GREEN

Welcome to the very first DVD from Linux Voice! This is a dual-layer 8GB disc, so you get twice as much data as in normal DVDs, and we've packed it with the best from the Linux and Free Software world. Along with big-name distros, we've also included the latest FreeBSD release for those who want to expand their skillset into other Unix flavours.

You'll also find some exclusive Linux Voice videos that we've made specially for this disc, showing some of the programs covered in the magazine. If you find them useful, let us know and we'll make even more of them!

We won't be having a DVD on every issue of Linux Voice, because some months are very quiet for distro and software releases. So our plan is to have a killer disc three or four times a year, gathering together the best distro releases, and rolling in much more as well. Oh, and if you're reading the digital version of Linux Voice, you can grab the DVD ISO image from our website at www.linuxvoice.com

Mike Saunders, Disc Editor
mike@linuxvoice.com

Desktop polish

Linux Mint 16 "Petra"

The latest and greatest from the world's most talked-about distro.

There seems to be no stopping Linux Mint. It has evolved from a simple fork of Ubuntu into a major project, with a bustling community and well-received releases. Mint takes the best of Ubuntu and Debian, mixes in its own software and adds a hugely polished interface on top. It's easy to install, easy to use and easy to love – and it's suitable for both new and experienced Linux users alike.

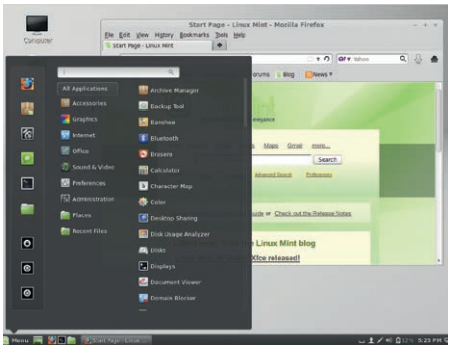
On this issue's Linux Voice DVD you'll find the new release of Linux Mint, codenamed "Petra". It's the 32-bit version with the Cinnamon 2.0 interface (which has gone through a lot of refinement since earlier releases). System requirements are:

- **512MB RAM**
- **5GB hard drive space**

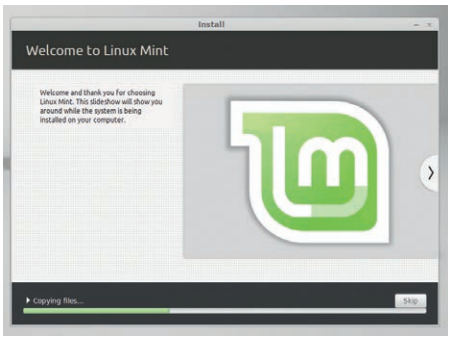
That will get you by, although we recommend a minimum of 1GB RAM for smooth running. To install Linux Mint, just boot your PC from the Linux Voice DVD – ie start your PC with the disc in the drive. (On some older PCs, you may need to change the boot order in your BIOS so that the DVD drive boots first, so consult your PC documentation or your local Google.)

Choose "Start Linux Mint" from the boot menu; if you have any problems booting, restart and choose the Compatibility Mode option. When you reach the desktop, you can explore the software in Live mode, or double-click the installer icon on the desktop to install Mint to your hard drive. If you're a new Linux user, chances are that your PC already has Windows installed – in this case, you can shrink the Windows section of your drive and install Linux Mint alongside it. Then, when you start your PC, you can choose between Linux and Windows.

Mint is one of the most user-friendly distros around, but if you need any help, visit **www.linuxmint.com/documentation.php** for a great user guide.



It's pretty, fast, easy to use and it has loads of apps – Mint is one of our favourite Linux distros.



Installing Mint to your hard drive only takes a few clicks, and you can keep Windows if you want a dual-boot system.



Developers' paradise

Fedora 20

Cutting-edge software with a lovely Gnome 3 desktop.

We weren't all big fans of Gnome 3 when it arrived: major features had been removed from the 2.x series of the desktop, and the new interface design was hard to understand. That was years ago, though, and Gnome 3 has come on in leaps and bounds. Arguably the best showcase of Gnome 3 has been Fedora, and from the LVDVD you can install the new release of this popular desktop distro.

Fedora 20 is bootable directly from the disc, so start your PC with the disc in your drive and choose it from the boot menu. (If you encounter graphics problems when starting up, reboot and choose the "basic graphics mode" option.) System requirements are:

- **1GB RAM**
- **10GB hard drive space**

We recommend a minimum 1.5GHz 32-bit or 64-bit processor. While Gnome is the default desktop environment, if you're on a low-spec machine you can install and switch to a lighter desktop such as Xfce.

Anyway, once Fedora has booted from the DVD, you'll be given the option to either use the distro in live mode or install it to your hard drive. If you choose the latter, you'll be given options for how you want to partition your hard drive: you can devote it entirely to Fedora, or you can install it alongside an existing Linux or Windows installation, giving you a boot menu when you start your PC.



If you hated earlier versions of Gnome 3, it's worth giving the desktop another chance.

Fedora has one of the most active communities of any Linux distro, so if you need any help or support, you'll be able to find help at **www.fedoraproject.org**.

Strap here strap here

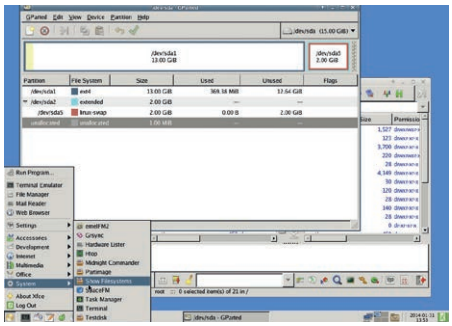
Other distros

Arch, FreeBSD, SystemRescueCD

Alongside the two big desktop distros in the shape of Fedora and Mint, we have some extras worth exploring. If you're looking for a power-user distro that's constantly up to date, try Arch. This has a rolling release model, where instead of getting a bunch of new software every six months or a year, you get the latest packages when the Arch team releases them. We've provided an ISO image of the latest installer – try our tutorial on page 92 to discover how best to install it.

Then there's FreeBSD 10, supplied in two formats for writing to a USB key or CD-R. The **README.TXT** file explains how to do the former, and we've also included the fantastic FreeBSD Handbook that explains all aspects of the OS (extract **book.html-split.tar.bz2** and open **index.html** in your browser).

Finally we have SystemRescueCD, a useful little distro that can get your PC out of tough times. You can use it to fix filesystem problems (both Linux and Windows), and it comes with various other recovery tools. On the LVDVD you'll find an image that you can write to a USB key or a CD-R.



SystemRescueCD can be a godsend for fixing broken machines – it's well worth keeping around on an old USB key.

And there's more!

Videos, podcasts, software, oh my!

There's much more on the DVD as well: we've made five videos exploring some of the software covered in the magazine, so you can, for instance, see the awesome new document collaboration feature in OwnCloud 6 without installing it. These videos are in Ogg Theora/Vorbis (.ogv) format, and will play directly in Firefox – so there's no need to install extra media players or codecs.

Then we have the latest Linux Voice podcasts. If you've never heard them before, we've been recording podcasts for several years now, and every two weeks we gather round microphones to chew over the big issues in Linux and Free Software. Expect a lot of opinions, a lot of rants, and a lot of off-topicness as well.

And then we have new software releases, FOSSpicks, games, the latest GCC and kernel source code (if you love to keep your base system bang up-to-date) and more LInux essentials. Enjoy exploring the DVD, and stay tuned for another bursting-at-the-seams disc in a few issues' time!



Our pub podcasts are especially popular – here we are at The White Hart in Calne.

/dev/random/

Final thoughts, musings and reflections



Nick Veitch was the original editor of Linux Format, a role he played until he got bored and went to work at Canonical instead. Splitter!

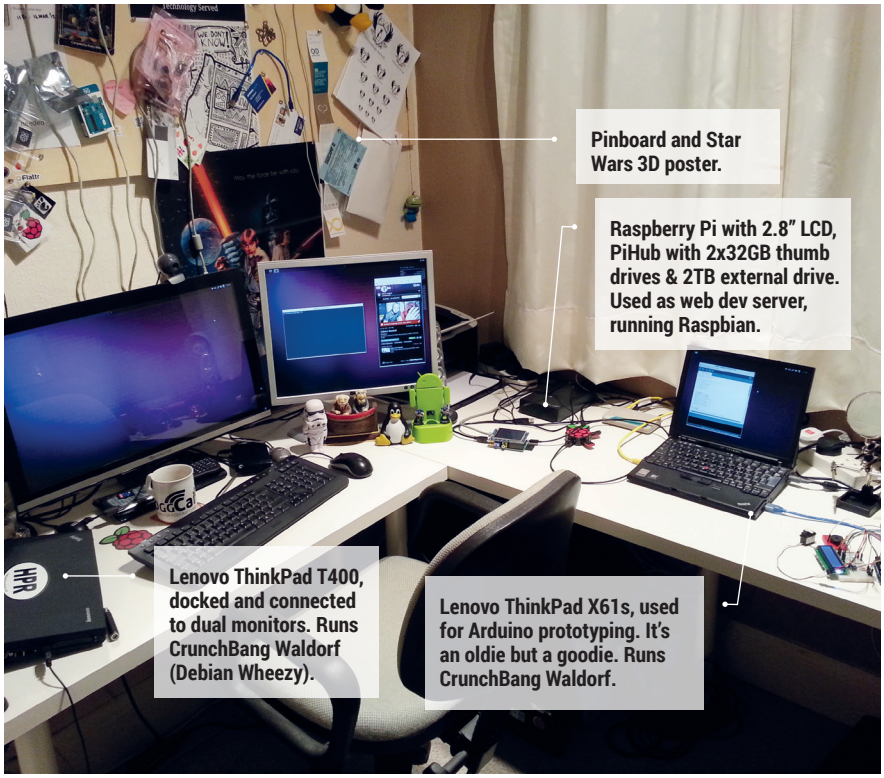
I had intended to write a sort of profit/loss account of open source for recent history – some small way of seeing where the big wins were for free software and where things could be better. But before I put fingers to keyboard, something happened to change that.

I came across something called Kafka, which is a cunning message filtering server – services publish messages to it, and listeners can filter on various tags or topics to receive a stream of relevant messages (**https://kafka.apache.org**). It is the sort of glue service that drives social media newsfeeds, or possibly anti-social media too if you wanted it to.

Kafka is cunning, but that isn't why it stopped me writing my profit and loss story. The startling thing about Kafka isn't that it's good software, but where it came from. Any guesses? Google? Some group of ambitious hackers? No, the Kafka project was founded and is maintained by that bastion of free software, LinkedIn.

Why is this useful piece of technology open sourced under the Apache 2.0 licence? Because people at LinkedIn recognised that it's a great way to get better code. That by allowing liberal usage and community review, they end up with better software to do the things they do. If other people also end up with better software, that's all to the good, and in fact helps test, refine and make the service the best available.

So, it sort of makes the whole profit and loss idea a bit redundant. I don't think anyone would claim that LinkedIn's dev team is comprised of starry-eyed idealists who sing the GNU song every morning before settling in to code under wall-sized portraits of RMS (but, you know, if you are reading, LinkedIn people, do tell. Send pics), which only makes it even more obvious that open source just simply makes sense.



My Linux setup Philip Newborough

The creator of Crunchbang Linux show us his den.

Q What version of Linux are you using at the moment?

A Debian Wheezy, in the form of CrunchBang Waldorf – I love the stability that Debian affords me.

Q What desktop do you use? We're guessing OpenBox, as that's what desktop Crunchbang uses by default.

A It depends – are we talking virtual or physical? Virtually, I use OpenBox, yes. Physically, I use a Lenovo ThinkPad X220 as my daily driver, but switch to my Lenovo ThinkPad T400 when I'm in my cave.

Q What was the first Linux setup you ever used?

A SUSE 6.3 in 1999, it was a PITA to set up, but also a lot of fun.

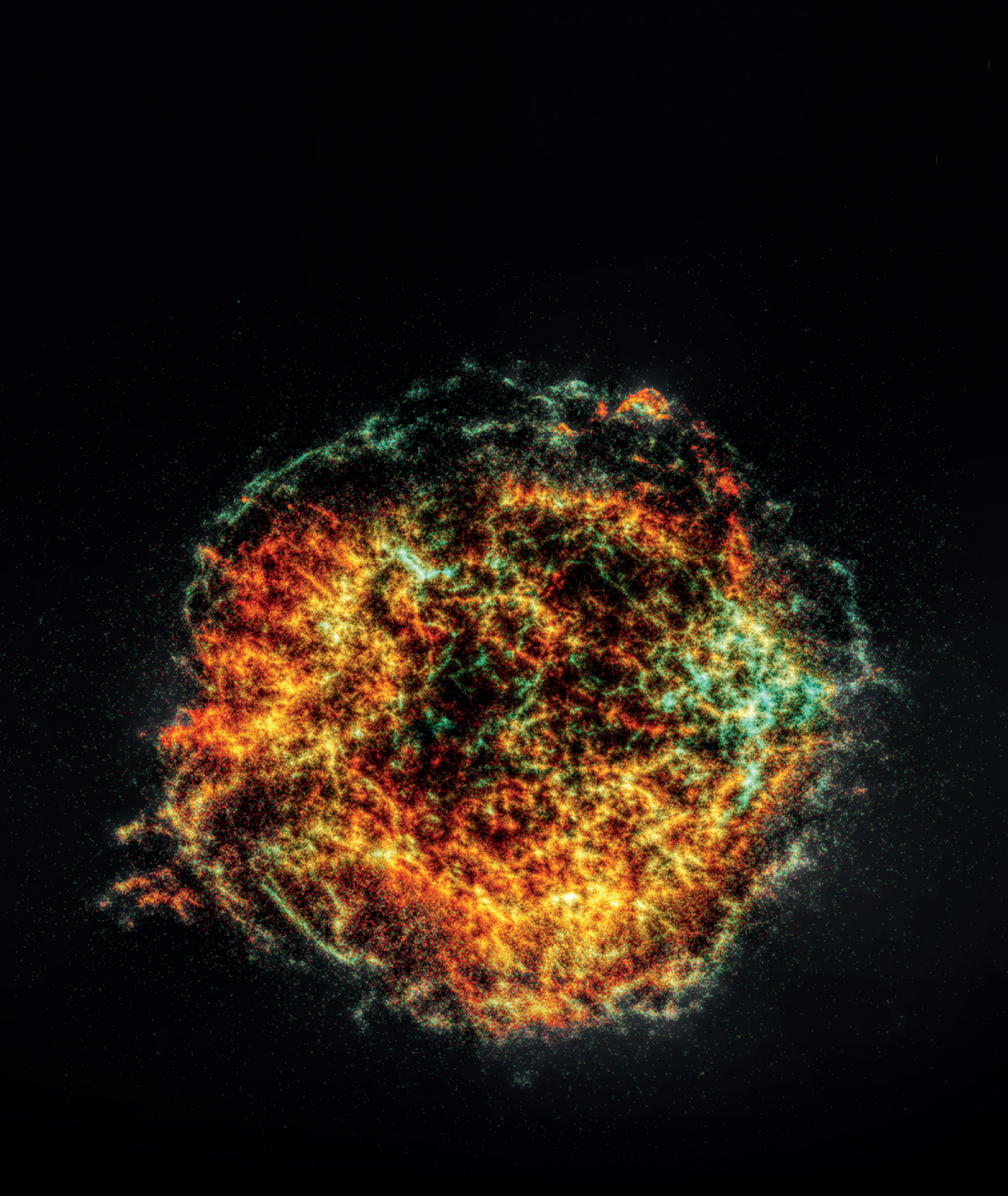
Q What Free Software/open source can't you live without?

A Tough question, I rely on so much. I automate/script a lot of tasks, so Bash, GNU Coreutils and OpenSSH would have to figure in the equation.

Q What do other people love but you can't get on with?

A I have no idea!

Follow us on Twitter @linuxvoice!



Supernova remnant Cassiopeia A
Data from Chandra X-Ray: NASA/CXC/SAO/D.Patnaude et al.
Contour mapped with Luminance-HDR 2.4.0


LV